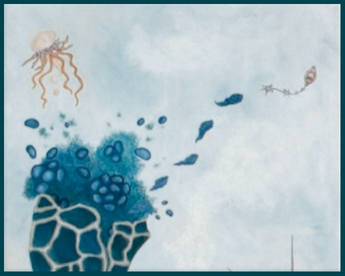
Proceedings of the European Conference on Artificial Life 2015



edited by

Paul Andrews, Leo Caves, René Doursat,
Simon Hickinbotham, Fiona Polack,
Susan Stepney, Tim Taylor and Jon Timmis

Proceedings of the European Conference on Artificial Life 2015

ECAL 2015

Edited by:

Paul Andrews, Leo Caves, René Doursat, Simon Hickinbotham,
Fiona Polack, Susan Stepney, Tim Taylor and Jon Timmis

Cover image: *Natural Complexity (2008)*, an original artwork created by members of the YCCSA community (Reidun Twarock, Julie Wilson, Tanja Miethe and Jenny Burrow) inspired by the work of Ernst Haeckel.

ECAL logo: designed by Jamie Marsden and Briony Thomas.

Table of Contents

INVITED CONTRIBUTIONS 1

SPECIAL SESSION: SLIME MOULD COMPUTERS 14

SPECIAL SESSION: MORPHOGENETIC ENGINEERING 41

SPECIAL SESSION: QUANTIFYING EMBODIMENT 68

MAIN ECAL TRACK 79

Preface

This volume presents the proceedings of ECAL 2015: the 13th European Conference on Artificial Life, held 20–24th July 2015, in York, UK (http://ecal2015.alife.org).

Since the first ECAL in 1991, the conference has been held at various European locations every other year, alternating with ALife (the International Conference on the Synthesis and Simulation of Living Systems). The history of those conferences and their past websites can be found on the website of the International Society for Artificial Life (ISAL, http://alife.org/).

We solicited submissions in two formats: full papers (up to 8 pages) and abstracts (1 page). The extended abstract format was implemented in 2008 (ALIFE 11) to promote participation of researchers working in those disciplines where conference submissions require abstracts only. This format may be used for presenting either original work or recently published work. Both full papers and extended abstracts were peer-reviewed by multiple Program Committee members. Authors could opt for oral or poster presentation in either category. We accepted 101 submissions for oral presentation and 12 submissions for poster presentation. Authors of accepted papers modified their final versions based on the peer-review feedback.

We instituted a 'late breaking papers' session, soliciting 1–3 page abstracts which were subject to a light touch peer review, offering participants the opportunity to present up to the minute results. (These papers are available in a separate document.)

We also instituted a 'bring a poster' session (separate from the formal poster presentation session), to allow people to advertise their ongoing projects and research.

The program of ECAL 2015 comprises of a number of scientific, professional, and creative activities, including:

- Plenary talks
  - Rachel Armstrong (Newcastle University, UK)
  - Katie Bentley (Harvard Medical School, USA), who unfortunately had to cancel her appearance, but her invited abstract is included in the proceedings
  - Paulien Hogeweg (Utrecht University, Netherlands)
  - Andy Lomas (Digital Artist, UK)
  - Justin Werfel (Harvard University, USA)
- oral presentation sessions (three parallel sessions) on a wide range of Artificial Life topics:
  - morphogenetic engineering (special session organised by Rene Doursat and Hiroki Sayama)
  - quantifying embodiment (special session organised by Keyan Ghazi-Zahedi, Christoph Salge, Georg Martius and Daniel Polani)
  - slime mould computers (special session organised by Andrew Adamatzky)
  - agent strategies
  - artificial chemistries
  - behavioural and evolutionary robotics

  - cellular automata
  - collective behaviours
  - communication and language
  - complexity
  - computational biology
  - ecosystems
  - evolutionary dynamics
  - in materio systems
  - in vitro systems
  - learning and evolutionary software systems
  - programming ALife
  - shape and form
  - social networks
  - swarm intelligence

- a poster presentation session

- satellite workshops

  - *CoSMoS 2015: the 8th Complex Systems Modelling and Simulation Workshop* (organisers: Paul Andrews, Susan Stepney)
  - *EPS: Evolution of Physical Systems Workshop* (organiser: John Rieffel)
  - *EvoEvo: Evolution of Evolution* (organisers: Guillaume Beslon, Santiago Elena, Paulien Hogeweg, Dominique Schneider, Susan Stepney)
  - *FoCAS: the 2nd FoCAS@ECAL Workshop on Fundamentals of Collective Systems* (organisers: Emma Hart, Ben Paechter; run in conjunction with SteerComplex)
  - *HieDy: Hierarchical Dynamics* (organisers: Jon Rowe, Chris Cannings, Peter Dittrich, Martin Nilsson Jacobi)
  - *ITIALIFE: Information Theory in Artificial Life* (organisers: Georg Martius, Christoph Salge, Keyan Ghazi-Zahedi, Daniel Polani)
  - OEE: Open Ended Evolution – Recent Progress and Future Milestones
    (organisers: Tim Taylor, Alastair Channon, Mark Bedau)
  - *PhyChip: Physarum Machines – Slime Mould Computing and Novel Computing Substrates* (organiser: Andy Adamatsky)
  - *SB-AI 2015: What can Synthetic Biology offer to Artificial Intelligence?* (organisers: Luisa Damiano, Yutetsu Kuruma, Pasquale Stano)
  - *SteerComplex: Steering Complex Systems* (organisers: Alexandra Penn, James Dyke; run in conjunction with FoCAS)
  - *SynNatSys: 1st Workshop on Synthetic Natural Systems* (organiser: Soichiro Tsuda)
  - *ToProB: Towards Programmable Biology* (organisers: Harold Fellermann, Omer Markovitch, Owen Gilfellon, Curtis Madsen)
  - *TRUCE: Unconventional Computation in Europe* (organiser: Martyn Amos)

- tutorials

  - *Aevol tutorial* (tutors: Guillaume Beslon, Dusan Misevic, Carole Knibbe, David Parsons)
  - *Evolutionary Robotics* (tutorrs: Nicolas Bredeche, Stéphane Doncieux, Jean-Baptiste Mouret)
  - *ICoNMAP: Introduction to Complex Networks Modeling and Analysis with Python/NetworkX* (tutor: Hiroki Sayama)

  - *Introduction to JIDT: An information-theoretic toolkit for studying the dynamics of complex systems* (tutor: Joseph Lizier)
  - *Physarum Machines: Slime Mould Computing and Novel Computing Substrates* (tutor: Andy Adamatsky)
  - *An Introduction to the use of Proper Statistical Analysis for ALife Experiments* (tutor: Mark Wineberg)

- The second ISAL summer school
  - *Historical and Philosophical Perspectives on Artificial Life* (tutor: Mark Bedau)
  - *Unconventional Computing* (tutor: Susan Stepney)
  - *Origins of Life and Protocells* (tutor: Steen Rasmussen)
  - *Introduction to the Modeling and Analysis of Complex Systems* (tutor: Hiroki Sayama)
  - *Autonomic Computing and Self-Adaptive Networks* (tutor: Ada Diaconescu)
  - *Model Driven Engineering and Biomedical Simulations* (tutors: Fiona Polack, Louis Rose)

We would like to thanks our sponsers, for their financial and in-kind contributions, without which ECAL 2015 would not have been a success:

- ELSI: Earth-Life Science Institute Origins Network
- FoCAS Project: Fundamentals of Collective Adaptive Systems
- HIERATIC Project: Hierarchical Analysis of Complex Dynamical Systems
- MIT Press
- SimOmics
- Springer

Any scientific conference like ECAL relies heavily on the hard work of its Programme Committee, who provide detailed comments on all the submitted papers. We thank this year's committee for their timely and professional reviews.

Programme Committee

Andy Adamatzky
Andreas Albrecht
Bradly Alicea
Fernando Almeida E Costa
Lee Altenberg
Martyn Amos
Paul Andrews
Takaya Arita
Joshua E Auerbach
Jacob Beal
Randall Beer
Katie Bentley
Peter Bentley
Guillaume Beslon
Navneet Bhalla
Martin Biehl
Eleonora Bilotta
Daniel Bisig
Tim Blackwell
Christian Blum
Josh Bongard
Terry Bossomaier
Amine Boumaza
Markus Brede
Nicolas Bredeche
David Breen
Micah Brodsky
Larry Bull
Seth Bullock
Lola Cañamero
Angelo Cangelosi
Timoteo Carletti
Pedro Castillo
Leo Caves
José M Cecilia
Martin Cenek
Sung-Bae Cho
Dominique Chu
Nikolaus Correll
Ernesto Costa
Sylvain Cussat-Blanc
Thomas Dandekar
Christian Darabos
Ada Diaconescu
Stéphane Doncieux
Alan Dorin
Rene Doursat
Alastair Droop
Stefan Dulman
Yves Duthen
James Dyke
Matthew Egbert
Harold Fellermann
Christoph Flamm
Gary Fogel
Tom Froese
Simon Garnier
Nicholas Geard
Philip Gerlee
Carlos Gershenson
Keyan Ghazi-Zahedi
Mario Giacobini
Jean-Louis Giavitto
Heather Goldsby
Ángel Goñi Moreno
Laura Grabowski
Robin Gras
Hugo Gravato Marques
Heiko Hamann
Taichi Haruna
Evert Hassdijk
Helmut Hauser
J. Michael Herrmann
Simon Hickinbotham
Thomas Hinze
Matěj Hoffmann
Paulien Hogeweg
Paul Humphreys
Phil Husbands
Genki Ichinose

Fumiya Iida
Hiroyuki Iizuka
Takashi Ikegami
Eduardo Izquierdo
Klaus Jaffe
Jerzy Gorecki
Yaochu Jin
Michal Joachimczak
Jeff Jones
George Kampis
Yoshihiko Kayama
Cristian Kellog
Carole Knibbe
David B. Knoester
Marcelo Kuperman
Yutetsu Kuruma
Hui Keng Lau
Joel Lehman
Tom Lenaerts
Lukas Lichtensteiger
Soo Ling Lim
Tiong Hoo Lim
Joseph Lizier
Daniel Lobo
Michael Lones
Robert Lowe
Herve Luga
George Magoulas
Georg Martius
Alexander Maye
Richard Mayne
Barry McMullin
Peter Mcowan
Jj Merelo
Daniel Merkle
Olivier Michel
Alan Millard
Julian Miller
Dusan Misevic
Shuhei Miyashita
Sara Montagna
Jean-Marc Montanier
Jean-Baptiste Mouret
Chrystopher L. Nehaniv
Surya Girinatha Nurzaman
Charles Ofria
Yuta Ogai
Eckehard Olbrich
Randal Olson
Jonathan Pascalie
Joshua Payne
Alexandra Penn
Jeffrey Pfaffmann
Macin Pilat
Fiona Polack
Daniel Polani
Simon Powers
Mikhail Prokopenko
William Ratcliff
Thomas Ray
Daniel Richards
John Rieffel
Sebastian Risi
Luis M. Rocha
Andrea Roli
Christoph Salge
Kazutoshi Sasahara
Hiroki Sayama
Thomas Schmickl
Jaekyun Shin
Tomohiro Shirakawa
Eric Silverman
Georgios Sirakoulis
Antoine Spicher
Russell Standish
Pasquale Stano
Susan Stepney
Kasper Stoy
John Sullins
Yasuhiro Suzuki
Reiji Suzuki
Uwe Tangen
Charles Taylor
Tim Taylor
Christof Teuscher
Serge Thill
Jon Timmis
Jim Torresen
Martin Trefzer
Vito Trianni
Soichiro Tsuda
Elio Tuci
Andy Tyrrell
Jon Umerez
Tatsuo Unemi
Edgar Vallejo
Sergi Valverde
Andrew Vardy
Neil Vaughan
Nathaniel Virgo
Mirko Viroli
Sebastian von Mammen
Aaron Wagner
James Alfred Walker
Justin Werfel
James Whiting
Lance Williams
Borys Wrobel
Andrew Wuensche
Lidia Yamamoto
Xin-She Yang
Luis Zaman
Hector Zenil
Xiaoge Zhang

Subreviewers

Arthur Bernard
Edgar Buchanan
David Buckingham
Matthew Dale
Emily Dolson
Adam Erskine
Cai Gao
Zhen Hu
Anya Johnson
Michael Krastev
Rebecca Naylor
Richard Redpath
Héctor Sánchez
Chris Timperley

We would also like to thank: April Denison, for her excellent help with the registration and accommodation site; and all the student helpers ‘on the ground’ at the conference.

We hope you enjoy, and are enlightened by, the research presented in these proceedings.

The ECAL 2015 Organising Committee:
Susan Stepney
Jon Timmis
Paul Andrews
Leo Caves
René Doursat
Simon Hickinbotham
Fiona Polack
Tim Taylor
Sarah Christmas
Bob French

Non-random random mutations: a signature of evolution of evolution (EVOEVO).

Paulien Hogeweg[1]

[1] Theoretical Biology and Bioinformtics group, Utrecht Univiersity, Utrecht, the Netherlands
P.Hogeweg@uu.nl

What has evolved in the ca 3.5 billion years of biological evolution on earth? A first answer which may spring to mind is your favorite organism, be it elephant, fly, slime mold, microbe or man. Other answers might be the stunning apparent diversity of lifeforms, or the greatly enhanced energy cycling at the earth surface. Or all this may be simplified to characterizing what has evolved as 'complexity'. The latter answer is often pursued within artificial life and is indeed possibly the easiest to study because the most abstract and least well defined, but certainly not trivial as the basic mechanism of Darwinian evolution leads to adaptation but not necessarily to complexity.

Beslon et al (evoevo-project http://www.evoevo.eu/) recently posed an other answer for the above question, namely that what has evolved is evolution itself, a perspective which is now pursued under the name EVOEVO. When we study in vivo or in vitro evolution of extant organisms, it is this evolved evolution that we observe. This in contrast to in silico evolution where we most often observe the course of evolution from random initial conditions.

The genomic revolution in biology allows a much closer look at evolution of evolved organisms than ever before. Some of the striking observation gained include:

- very fast adaptation (involving few mutations) to environmental changes or genome changes (knock-outs).
- adaptive mutations are often (further) deletions.
- Last common ancestors of lineages at multiple scales are surprisingly large.
- blow-up of single point mutation (SNP's) to expression changes of multiple genes.
- many processes which cause and regulate changes in DNA have been observed.

The last point has been has been stressed by Shapiro in his book 'Evolution: a view from the 21st century', in which he indeed tries to convince the reader with numerous examples that the premise "random mutations" is the basic fallacy of evolutionary theory. Despite the title of the book he does not discuss the evolution of such mutational mechanisms, which in my view is the basic fallacy of the book. Indeed elucidating how such non-random mutations evolved by the basic process of Darwinian evolution by random mutations and selection is in my view the challenge we want to tackle in EVOEVO. We [1] have made some progress in this direction, although we are only at the beginning of this pursuit.

Surprisingly some of these surprising features of evolution of evolved organisms turn out to be generic properties of Darwinian evolution. They had not before been recognized as such because both population genetics and in silico evolution protocols often restricts the degrees of freedom of the evolutionary process. For example, in multilevel evolutionary models, we have shown that early genome inflation, followed by fitness gain by deletion mutation is a generic pattern in the evolutionary history of *successful* lineages, (i.e. those lineages we should expect to encounter).

"Non-random mutations" in the broad sense involves both biased and/or regulated changes in the genome, as biased effect of mutations at the phenotypic level. We have shown that both these types of non-random mutations evolve in variable environments in a strict Darwinian setting. For example transposon dynamics can structure genomes such that beneficial mutations are overrepresented and very fast evolutionary adaptation to alternative environments evolve. Another example is the evolution of high HGT rates for particular types of genes. However non-random effect of random mutations seem to be an even stronger mechanism to enhance evolvability in evolved organisms.We have shown a bias to beneficial effects of random mutations of all types in simple models where the effect of random mutations is mediated by GRNs and/or metabolism. This is true to such an extend that such enhanced evolutionary potential seems to be a viable alternative to physiological (GRN based) adaptation to changing environments.

All these cases can be understood in terms of the evolution towards a very specific mutational neighborhood (and therewith the evolved quasi-species) through shaping the genome structure, the mutational processes and the genotype-phenotype mapping over long term evolution.

[1] in particular Anton Crombach, Thomas Cuypers, Folkert de Boer, Nobuto Takeuchi, Sandro Colizzi and Bram van Dijk contributed to the results discussed here.

How do the origins of life sciences influence 21st century design thinking?

Rachel Armstrong

School of Architecture, Planning and Landscape, Newcastle University, UK, NE1 7RU
rachel.armstrong3@ncl.ac.uk

Abstract

How can the origins of life sciences inform design thinking in an ecological era?

This paper considers the possibility of the origins of life sciences as being more than a blue sky practice for the development of advanced scientific theories but also offers a technical platform for designing and engineering life-like solutions for an ecological era. A design-led study of dissipative systems is discussed as a form of natural computing and innovation platform that can deal with probability and whose ontology (nature of becoming) is consistent with complexity, nonlinear dynamics and the flow of energy and matter. However, since the proposed approaches do not yet formally exist as products or mature technologies, exemplary design-led projects are introduced to explore the principles of design and engineering with these origins of life strategies. A portfolio of work is presented that includes the The Hylozoic Ground installation and Future Venice projects. Such experimental work investigates the value of collaborations between the origins of sciences and design practice as a strategic approach towards new systems such as, artificial soils – which may not only be recognised as applied research fields that offer insights into the transition from inert to living matter – but also give rise to potential cultural impacts and commercial opportunities in the built environment.

Introduction

We are in the midst of a transition from an industrial to an ecological paradigm of practice.

This is not as simple as making a substitution of an object-centered view of reality and supplanting it with process, complexity, networks and nonlinearity. Rather, it involves constructing a framework for understanding a world in continual flux that is navigated by many overlapping models of thought. This brings about paradoxes in the way that we work and solve complex challenges. The role of design in this quagmire of choices is to provide navigational avatars that can deal with inevitable inconsistencies when these models overlap, so that we may shape new values and forge new cultural practices.

At the heart of this transition is the appreciation that the world is in constant flux and that the matter from which it is formed – is lively. Responding to grand global challenges such as, climate change, increasing population density and the sustainability of cities, architects and designers have been looking for new ways of working with a whole range of strategies to counter the net effects of global scale, intensive industrial practices that are effectively reverse-terraforming our planet. Insights from the origins of life sciences point towards new opportunities in design thinking. Its quest is invested in the transition from inert to living matter – a complete reversal of the industrial paradigm.

Origins of life principles have actually pervaded design thinking with the rise of modern computing. Approaches that may be likened to information first strategies engage with many computer-modelling tactics, such as genetic algorithms and parametric design strategies. They generate a range of selectable objects that aim to resolve immanent forces within dynamic processes, which for example, have been applied in the work of William Latham (Latham, 1988) and Greg Lynn (Hight, 2007, p.43).

Metabolism first approaches are a more recent addition to the design portfolio and work directly with material self assembly and programmability, such as Autodesk's Bio Nano Programmable Matter research group that includes for example, Skylar Tibbits' work with 4D printing (Wakefield, 2013).

While origins of life sciences are generally regarded as 'blue sky' experimental practice with little, if any opportunity for technological innovation, recent collaborations between experimental architectural design and scientific research groups have opened up a potential range of applications with possible commercial value. While these cross-disciplinary approaches may still be some years away from becoming productised, a new portfolio of design language is emerging with an original ontology and epistemology that considers life itself as a technical platform and may fundamentally change our expectations of living systems.

Natural computing and origins of life sciences

This paper discusses a developing portfolio of design strategies within the field of natural computing to exemplify these approaches. Natural computing is a term that has been inspired by Alan Turing's interest in the technological potential of the natural world and consists of a range of overlapping scientific practices that range from the computer modelling of biological systems to working with programmable matter. It also shares many overlaps with morphological computing (Pfeifer and Iida, 2005) and unconventional computing (Adamatzky et al., 2007), which are also concerned with how matter makes 'decisions'.

Dissipative structures may be regarded as operative agents that can complexify space and matter. They function as an iterative system – or 'natural' counting process – that can be ordered to generate a range of programmable outputs. Jeremy England's notion of dissipative adaptation, as a possible complementary driver to evolutionary processes than genetic modification, is of particular interest as a generative and selective strategy when considering the transition from inert to living matter (Wolchover, 2014).

Dissipative structures arise spontaneously in nature across a range of scales such as, crystals, tornadoes and galaxies. While not all dissipative systems meet the technical qualifications of 'life' – all 'life' is a dissipative process. Those systems that are not given the full status of being truly alive are of great interest in the design process as a set of tactics that may increase the probability of life-promoting events. Potentially, when these spatialised material exchanges reach a certain degree of complexity, lifelike events may even be inevitable. Such substances sound mysterious, or even fictitious, but we can recognise them in everyday materials such as soils – on which all terrestrial life is founded. Soils represent an alternative material organising system than is used in our design processes today. Rather than being purified, homogenised and constrained within bounded spaces, they are open, messy and highly heterogeneous. This is of extreme interest in identifying processes that increase the liveliness of space and even raise the threshold of events that may spontaneously produce 'life' within a specific environment.

A series of design-led laboratory experiments demonstrate how such discursive systems may be constructed and provide a point of reference for thinking about how environments that support lively events – as opposed to designing life as a set of objects – may be approached as a way of producing 'artificial' soils that underpin fertile terrains, in which the probability of life is increased, using a set of simple techniques.

Generating dissipative structures

Dissipative structures may be produced at the intersections of mutually complexifying, active fields. This can be experimentally demonstrated using simple kitchen ingredients that provoke dynamic interactions. For example, water affinity can be used to produce a very simple dissipative system by harnessing the energetic and material potential that exists between substances that have very different relationships with water namely: glycerin that strongly attracts water and olive oil, which repels water. A base layer of glycerin and a top layer of olive oil create an interface at which droplets of food colouring are introduced using a hand-held pipette. Adding rock salt to the droplets increases their affinity for water, which provides further complexity and tension in the system. The hygroscopic properties of glycerin eventually overcome the osmotic pressure of the food colour droplets, at which point they explode downwards leaving structural trails in their wake that outline the resultant transient dissipative structure. Oil soluble coloring spreads sideways through the olive oil interface providing a residue of molecular activity – or, three-dimensional painting. Figure 1 is an installation for the On Architecture Exhibition at the Department of Arts and Sciences in Belgrade where food colour droplets have discharged into the base layer of glycerin leaving spidery threads of salt crystals behind. The technique is being prepared for use in secondary schools as a 'poor man's chemical garden'.[1]

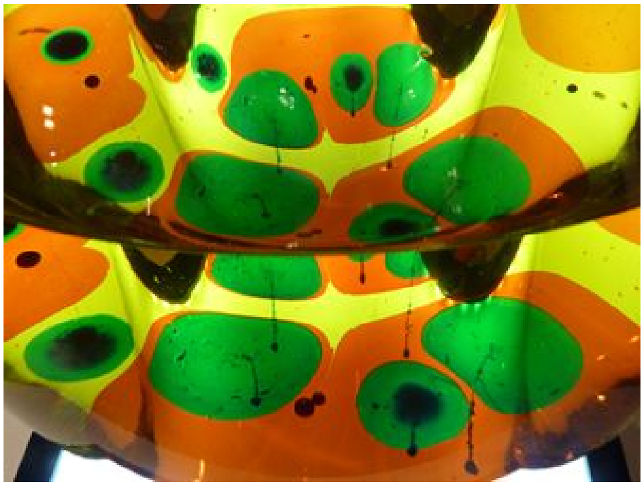


Figure 1: [Photograph by Rachel Armstrong, Academy of Arts and Sciences, Belgrade November 2013.] Oil and glycerin interface with table salt saturated droplets of food colouring. Exploded droplets leave crystalline trails as evidence of transient dissipative structures that emerge from the overlapping fields of activity in the experimental set up, which are competing for water availability.

Towards increasing complexity

Once potentiating fields of interaction have been established, the propagating waves of interaction produced by the os-

[1] Concern over the carcinogenic effects of crystalline salts, such as nickel and chromium, are becoming problematic for health and safety regulations. This 'edible' system therefore provides a highly colourful demonstrator for self-organisation without these regulatory challenges.

cillators begin to radiate and collide with each other. In reaction-diffusion computing, wave collisions have been used as tactics for establishing classical engineering strategies such as, logic gates (Adamatzky et al., 2005). Essentially this approach looks for events that can be translated and worked into mechanical circuits. In other words, complexity is being de-complexified so that it may be assimilated within a familiar mechanical engineering framework.

But what if an open-ended set of events within an environment are encouraged by starting at a rich and complex initial state as a way of preventing, or delaying the dissipative system from falling into a simple state – i.e. reaching equilibrium. Ikegami and Hanczyc (2009) call this approach the Maximalism design principle. The concept invites today's experimentalists to design more complex initial species of dynamic agents, where 'life' emerges at the edge of self-organisation and complexity. This theory is supported by England's observation regarding the dissipative adaptation of matter – that when a group of atoms is driven by an external source of energy (sun, chemistry), it will often gradually restructure itself in order to dissipate increasingly more energy. This could mean that under certain open environmental conditions, matter inexorably acquires the key physical attributes associated with life – and 'half-living' systems may emerge where bundles of self-assembling, mutually influencing bodies produce the conditions, and environments, of their own existence. Such complex events can be observed using Bütschli droplets, which represent an example of highly complex initial agents whereby each body influences its surroundings. The images in figure 2 are taken from a series of over 300 replicate experiments whereby lifelike events and organic structures with a range of striking morphologies and behaviours were produced within seconds to many minutes (Armstrong and Hanczyc, 2013).

Figure 3 shows an 'oceanic' ontology (Lee, 2011) of Bütschli species grouped according to observed morphological and behavioural typologies. The diagram represents the contextualisation of meta events between droplets with time within a complex field of activity. The stage is not a single reading of events but reflects multiple possibilities where the field of activity, is constructed through exploratory, graphical approaches. The resultant diagram maps relationships in the system rather than invoking the classical 'tree' metaphor of classification systems, which focuses on differences rather than similarities between actors. The graphic is centred at time zero, from which concentric circles radiate, representing an exponentially increasing series of time intervals. It depicts the intense self-organising activity that happens early on in the chemical reaction. An estimated 90% of chemical activity is completed within five minutes of activation of the system, although individual droplets have been observed to be active as long as an hour after their genesis. A spiral that represents complexity also radiates from the origin around which various droplet morphologies and behaviours that indicate change in the system are grouped subjectively. For example, 'oyster chains' are distinct in appearance but only differ from 'complex marine landscapes' by the relative degree of solid material they produce. The range of different bodies produced by the Bütschli system generates a potential portfolio and experimental starting point for 'soil-producing' technologies – fabrics and materials that may increase the material complexity and 'fertility' of a landscape and therefore the probable occurrence of lifelike events. Such systems do not have any direct commercial applications but do begin to suggest ways of taking a first-principles approach towards developing 'artificial' soils, or other life-promoting synthetic media.

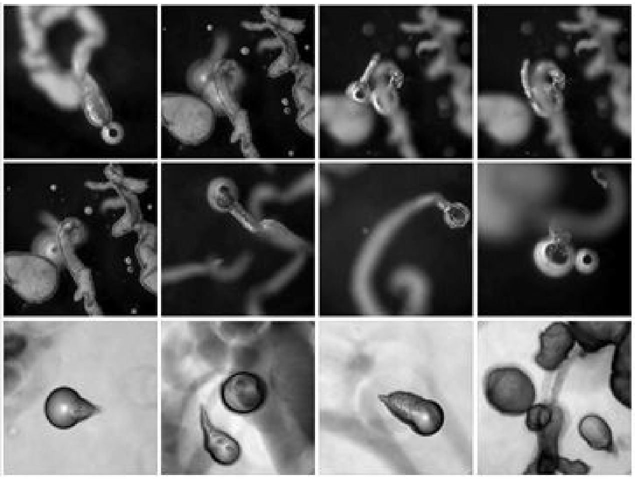


Figure 2: [Collage of movie stills by Rachel Armstrong, Southern University of Denmark, Odense, June 2009.] Range of osmotic structures produced by the Bütschli system.

Natural computing constitutes a platform that can facilitate entirely new design and engineering opportunities. These take the form of material convergences that exist within the entanglements of the spontaneous flow of energy and matter. Some of the lively bodies form loose, reversible groupings – or assemblages – with each other, while others become coupled and transformed by the interactions. Natural computing techniques also help break down some of the ontological barriers that set objects and systems in opposition and facilitates technological convergence whereby lively bodies may reach tipping points that produce radical breaks and discontinuities in the system.

Questions of scale

Unmodified Bütschli droplets spontaneously arise at the millimeter scale and last for many minutes. However, they can be easily made bigger and scaled to the centimeter scale as well as last longer, over a course of weeks, by adding an extract of their waste products to their body. Although these

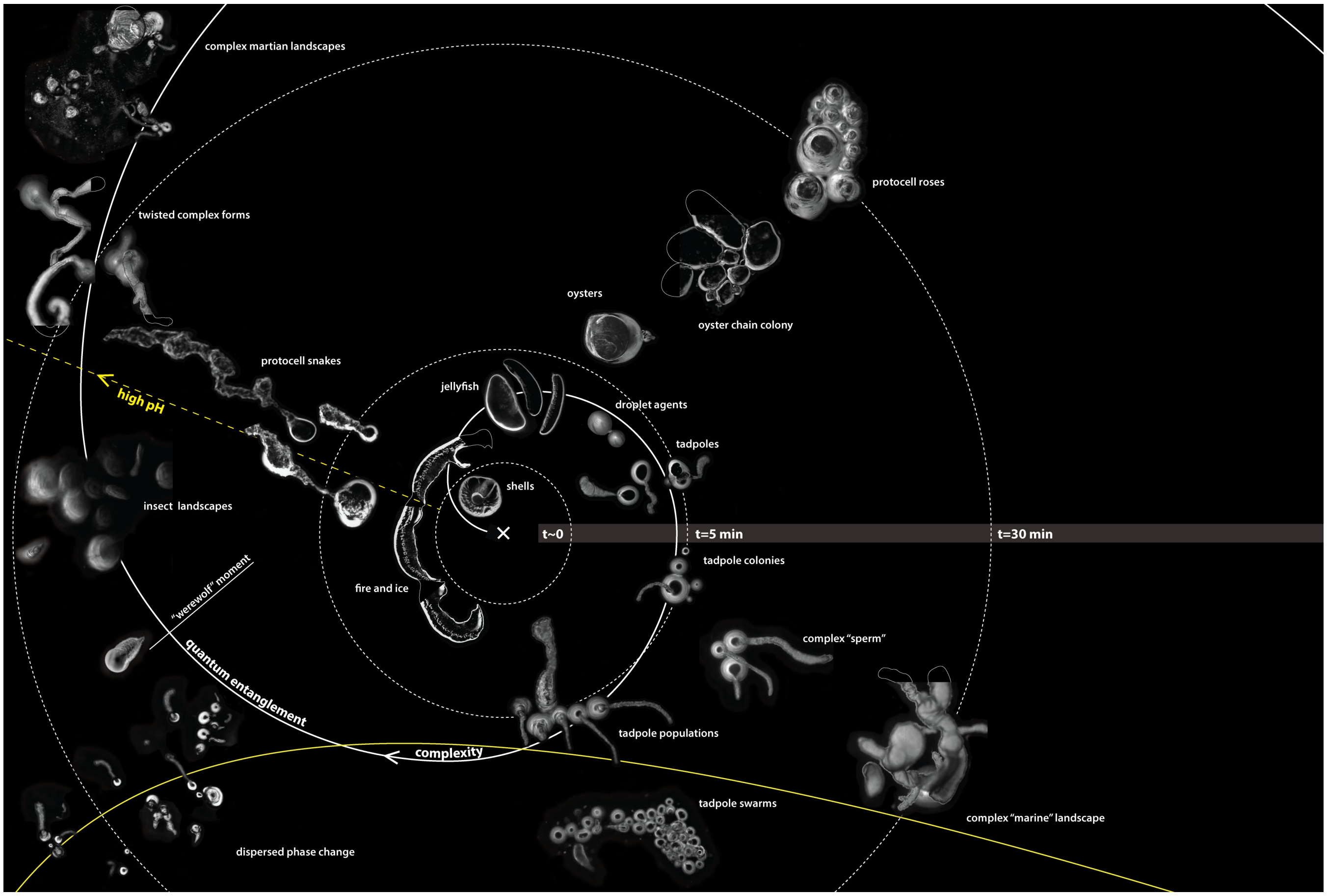


Figure 3: [Diagram designed by Rachel Armstrong and drawn by Simone Ferracina, July 2012.] This diagram depicts dynamic droplets as 'actors' that operate within the many variable influences encountered in their oil field as an ontolological 'map' of events. While the diagram is drawn as a 2D topology, the field events are manifold and open up multidimensional spaces through their continuous interactions that shape the evolution of the system.

agents are less lively than their original counterparts, they still possess the fundamental properties of dissipative systems. For example, in this attenuated form, they can be programmed to produce microstructures by adding salt solutions that transform soluble carbon dioxide into an insoluble carbonate precipitate (figure 4).

Physical constraints placed on the system can provoke a range of phenomena. For example Turing bands can be produced by reducing the physical dimensions of the reaction space to around 2 cm, which approximates with the maximum diameter possible for a modified droplet. These undulating configurations are the consequence of reaction–diffusion systems, which Alan Turing believed underpinned pattern formation in animals during morphogenesis, like dappling (Turing, 1952). This transformational ability was clearly demonstrated in this installation in Vienna at the Natural History Museum (figure 5). This design-led experiment suggests how environmental constraints can shape the outputs of complex self-organising systems and requires designers to consider the scale of operations and site-specificity of these systems.

Natural computing design programs

Once established, dissipative systems like the Bütschli droplets may be manipulated within a network of interactions between bodies by altering their internal and external conditions. However, such distinctions are over-simplistic as dissipative structures are inherently leaky and introduced events eventually influence the whole of a field through reaction-diffusion computing processes that can be shaped at multiple interference points.

Internal droplet modification

Bütschli droplets can be designed to create a range of different products – or secondary forms – by adding different chemistries to their internal environment. For example, microstructures may be produced at the oil/water interface by

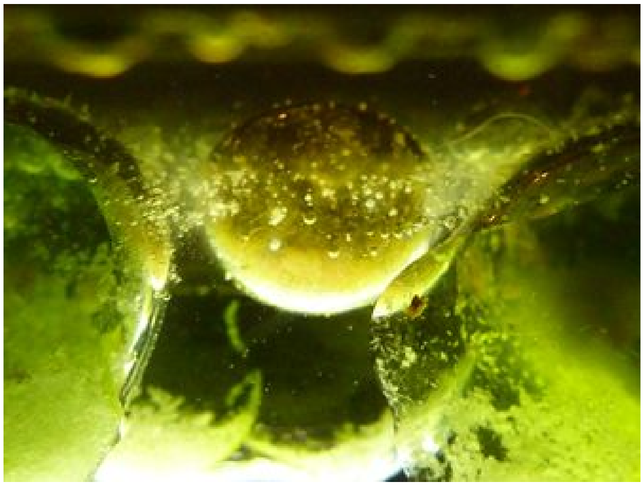


Figure 4: [Photograph Rachel Armstrong, Southern University of Denmark, April 2011.] Modified Bütschli droplets produce complex structures in the presence of mineral solutions.

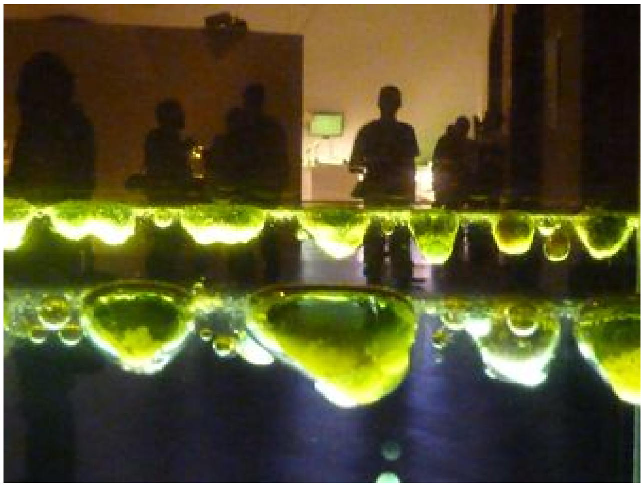


Figure 5: [Photograph Rachel Armstrong, Natural History Museum, Vienna, April 2011.] Modified Bütschli droplets exhibit Turing band pattern formation.

the formation of insoluble, magnetic 'magnetite' crystals by adding iron II and iron III salts (Berger et al., 1999). These design-led experiments produce sculptural inclusion bodies that can be used to indicate change over time in systems and have been used in installation work such as The Hylozoic Ground (figure 6).

External environment modification

Changing the external chemical conditions of the medium can also alter the behaviour of the Bütschli system based on physical and chemical changes, such as surface tension and chemotaxis (Toyota et al., 2009). For example, adding ethanol, or 'alcohol' to the olive oil field, produces a rather a dramatic effect characterised by the population-scale, sudden movement of the Bütschli droplets towards the alcohol source (figure 7). This may be explained by changes in surface tension that promote movement of the droplet dynamics, but also by reducing the viscosity of the olive oil (Armstrong, 2015).

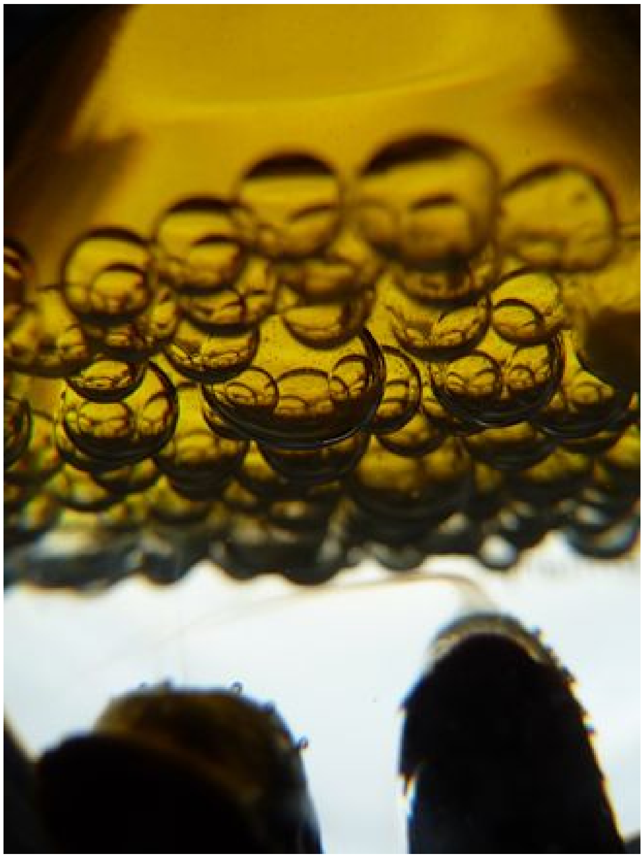


Figure 6: [Photograph Rachel Armstrong, Southern University of Denmark, Odense, June 2009.] Modified Bütschli droplets produce magnetite bodies that move through the droplet and produce stalagmite-like osmotic structures when they come in contact with iron II and iron III salts in a ratio of 1:2.

Population scale interactions

Perhaps surprisingly, dissipative systems are remarkably predictable, although they produce a spectrum of outcomes that operate within 'limits' of possibility, which are imposed by the properties of the system and its contexts (figure 8). Dissipative structures are highly resilient and able to transform themselves – structurally and morphologically – according to changes in their interior and exterior environment. They can also form reversible groupings that generate the life-like behaviours, which account for their flexibility, robust-

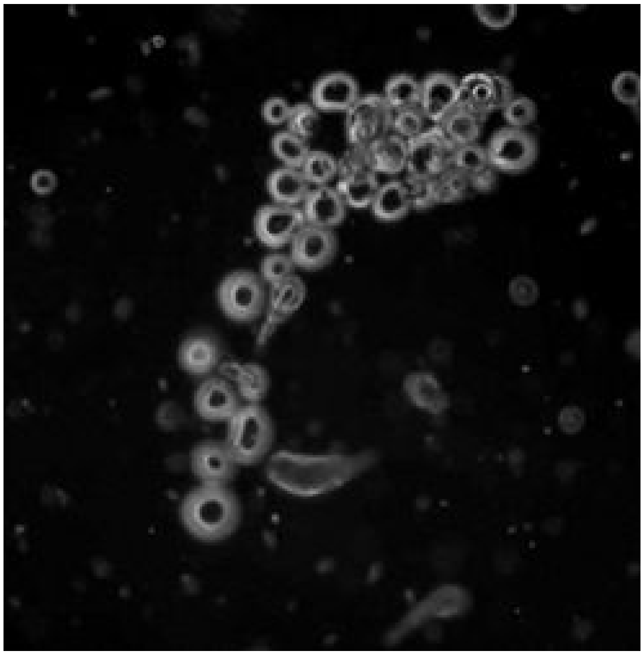


Figure 7: [Movie still Rachel Armstrong, Southern University of Denmark, Odense, June 2009.] Bütschli droplets respond to the presence of ethanol by clustering around the site of its addition to the olive oil medium.

ness and environmental sensitivity. However, populations of dissipative structures may periodically behave unpredictably when a tipping point is reached and give rise to novel, emergent, complex events that cannot be deduced from their characteristic behaviours (figure 9). New ways of accurately describing what is happening during striking phase changes are needed to more fully describe the continuous nature of change and its material complexity. Currently phase changes are described according to a set of recognisable meta-patterns that are documented over a series of time intervals. The relational aspects of tipping points therefore escape comprehensive description and analysis because of the way the system is observed during these events.

Elemental infrastructures

Enabling flow

Life-promoting elemental media are essential in facilitating the dynamic exchanges that allow dissipative structures to persist and evade the decay towards a disordered state, namely: air, heat, water and earth. Such infrastructures help optimise the conditions of maximalism and dissipative adaptation, whereby replenishing environmental media assists in producing complexity and dissipating heat in a self-organising system. Such infrastructures that form the basis of our water and nutrient cycles are likely to be critical in making the transition from inert materials towards lifelike

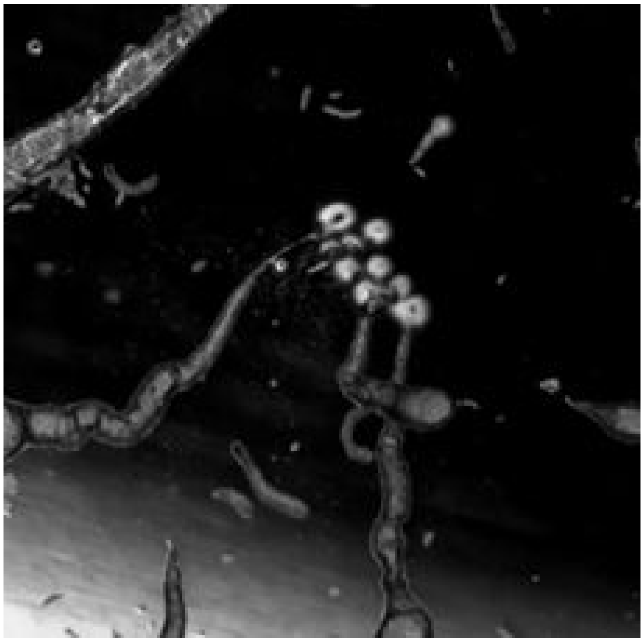


Figure 8: [Movie still Rachel Armstrong, Southern University of Denmark, Odense, June 2009.] Spontaneous assemblage of Bütschli droplets producing osmotic structures.

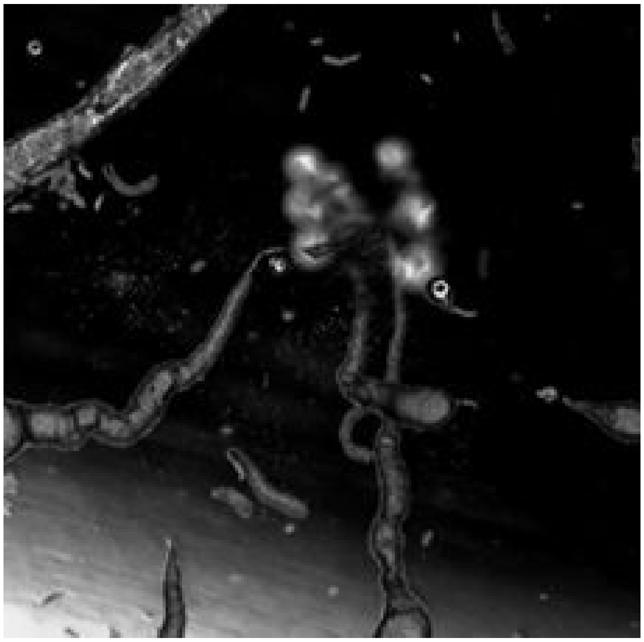


Figure 9: [Movie still Rachel Armstrong, Southern University of Denmark, Odense, June 2009.] Spontaneous phase change in morphology and behaviour in an assemblage of dynamic droplets that reach an unknown chemical ‘tipping point’ in the system.

systems.

Hylozoic Ground

The dissipative structures designed for Philip Beesley's Hylozoic Ground installation at the 2010 Venice Architecture Biennale were supported by a primitive infrastructure of elemental media. This jungle-like, room-sized, cybernetic, interactive mechanical matrix detected gallery visitors through an array of sensors that were coupled to effectors through a primitive neutral network. On activation the system shivered and raised friendly rubbery fronds of spikes as a somewhat disconcerting but entirely harmless greeting (figure 10).

Modified Bütschli droplets were installed as the dissipative system of choice, which could persist for the full 3-month duration of this exhibit where sustenance was provided through an arrangement of flasks containing Venice lagoon water that was open to the gallery atmosphere. Here, Bütschli droplets were entangled with the cybernetic system and its environmental interactions by responding to the presence of carbon dioxide through the production of tiny, brightly coloured structures about the size of a little fingernail that grew through the support provided by the liquid media in the flasks. Walking through this installation might be likened to exploring inside a giant nose – where the plastic fronds act as sinus hairs and the 'golden apple-like' flasks are 'smart' snot glands that can 'smell and taste' the presence of carbon dioxide. The Hylozoic Ground flasks were open to the air, and so enabled the on-going exchange of carbon dioxide across the air/water interface and increased the synthetic capacity of the modified Bütschli droplets.

Future Venice

Natural computing techniques are explored at the urban scale in the Future Venice project. The foundations of the city were bestowed with lifelike qualities by equipping them with a system of dissipative structures capable of many acts of synthesis through their interactions with the lagoon water. In turn, these designed metabolic networks enable the urban fabric to literally fight back against the damaging effects of natural elements in a struggle for survival – and therefore secure its longevity.

The dissipative system took the form of a series of dynamic droplets with a range of different chemical programs. Their metabolic interactions were demonstrated in the laboratory and experimentally designed to 'converse' with the lagoon environment. The combined interactions of these programmable droplets was to produce an accretion technology that is mediated at the interface between the lagoon water and the dynamic droplets. The watery infrastructure of the city provides the specially engineered droplets with an abundant flow of nutrients such as, dissolved carbon dioxide and minerals, whereby the collective action of the droplets forms an artificial garden reef underneath the city's foundations. This gradually creates a solid structure that spreads

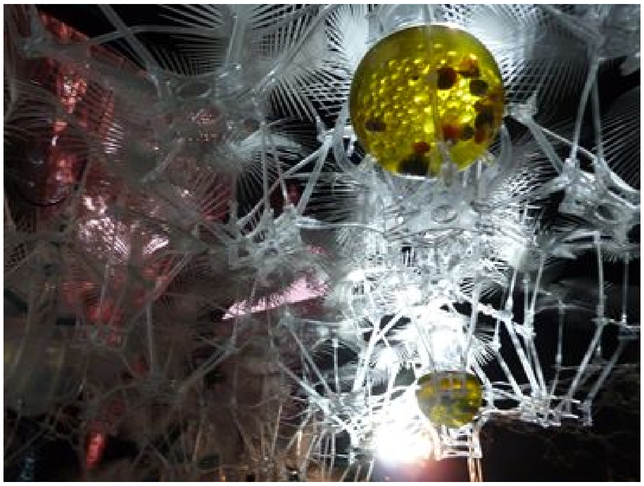


Figure 10: [Photograph Rachel Armstrong, Canadian pavilion, Architecture Biennale, Venice, August 2010.] The Hylozoic Ground installation is a cybernetic matrix that integrates a range of different dynamic chemistries to aesthetically and functionally complement the dynamic processes that inform architect Philip Beesley's installation. Here, chemical organs containing modified Bütschli droplets are open to the air and produce solid carbonate microsculptures in response to the presence of carbon dioxide.

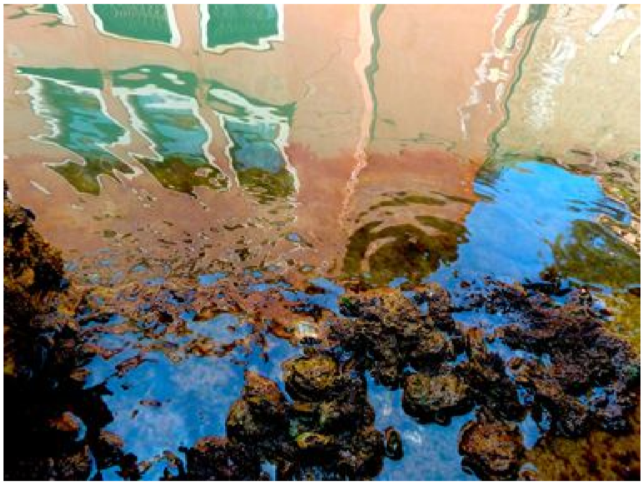


Figure 11: [Photograph Rachel Armstrong, Canalside Venice, August 2010.] Stromatolite-like formations in the Venetian canals are shaped over time to produce mineralised materials.

Venice's point load over a much broader base. Consequently Venice is prevented from sinking so quickly into the soft mud that it has been founded on (figure 11). A natural version of this accretion process is observed around the lagoon side and the canals, which is orchestrated by the native marine wildlife. Potentially, dynamic droplets could work alongside these organisms to co-construct an architecture that is mutually beneficial to the marine ecology and the city.

Importantly, should the environmental conditions change and the lagoon dries out – say for example, Pietro Tiatini and his colleagues succeed in anthropogenically lifting the city by pumping seawater into its deflated aquifers (Teatini et al., 2011), or if when the MOSES gates are raised in 2014 the native ecology reaches a catastrophic tipping point (Water-technology.net, 2015) – then the natural computer, which consists of the Venice waterways and the smart droplets, can follow a different program and re-appropriate its actions. Specifically, as the waters subside droplets coat the woodpiles in a downwards direction with a protective layer of 'biocrete' that stops them rotting when they are exposed to the air – instead of spreading outwards to form a reef.

Potentially natural computing processes could be applied to the whole bioregion of Venice. Facilitated by flowing water, the right kinds of metabolisms and spatial programs could give rise to tactics that generate, new relationships between natural and artificial agents. These may become the bedrock for forging life-promoting, synthetic ecologies. In this context natural computing enables a constant flux between fabric, space, structure and location. The outputs of the system do not imitate Nature but work as an alternative kind of life-promoting system – using a common chemical language based in physics and chemistry that is shared with the natural world. The idea of self-assembly in the production of materials and life-like architectures creates a potential commercialisable portfolio of solutions to deal with rising sea levels in coastal areas, bio-compatible materials, architectures that can deal with wet conditions and self-repairing systems. None of these possibilities have been formally productised but are informing further experimental research.

Yet, the design-led experiments that inform Future Venice do not propose to be a complete solution for the city's precarious future – or indeed erase our legacy of environmental woes. They do not attempt to 'solve' the inevitable changes that accompany a lively environment but open up the possibility of new approaches in addressing complex environmental events. Such dynamic practices generate new possibilities for metropolitan environments beyond the city of Venice. When adequately perfused by elemental infrastructures, they enable designers and engineers to regard urban landscapes as sites for pulsating, vibrating, transforming, flowing materials that may produce new kinds of experiences and spaces for innovation and inhabitation.

Building material complexity

To sustain the liveliness of an environment, a network needs to be prevented from reaching equilibrium. Technologies such as, continuous flow systems facilitate the persistence of dissipative structures (Graziani et al., 1976), where the increased complexity provided by these enriched environments also attenuates the participating bodies from reaching thermodynamic equilibrium. The infrastructural design of a dissipative system is therefore as important as the choice of oscillator system within it.

While dynamic droplets are supported in a free and open elemental medium like water, gels provide a matrix where water can move through structure. Gels provide the substrates for experimental systems that help investigate the kind of infrastructural support to facilitate the free flow of water, matter and air through the space. While their structure is nowhere near as complex as a cell milieu, or terrestrial soils, they create a set of conditions, such as diffusion and gradient formation, which provide an open system for experimenting on persistent dissipative structures.

Of particular interest in this context, Liesegang rings can demonstrate the generation of complexity within gels – a phenomenon that was discovered in the 19th century in developing photographic plate technology (Liesegang, 1896). The system is produced by soluble salts introduced into 'activated' (alkaline) gels that move sequentially as soluble and insoluble complexes through the matrix under the influence of gravity. The complex chemistries produced through activated gel systems were visualised in the Hylozoic Ground installation through Liesegang ring plates. These specially constructed narrow containers marked the passage of chemical time by the coupling and clustering of their dynamic chemical interactions.

Gel-based infrastructures facilitate natural computing strategies to generate increasingly complex interfaces that, through a range of differential physical and chemical phenomena, are rolled and folded in time, like embryonic plates. They demonstrate the multi dimensionality – or spatial nature – of natural computing that is also shaped through its historical events and contexts. These design-led experiments highlight the need for further research and development in the understanding of the development of dynamic elemental media and how these may be structured within a range of contexts – such as in under-imagined spaces within buildings like facades, cavity walls and underfloors – in which lifelike technologies and events may be desired e.g. in the production of chemical heating systems through composting processes.

'Artificial' soils as a potential origins of life sciences research discipline

While gels may prolong the actions of natural computing for days, or even months, persistent complex exchanges within the structure of matter are most successfully embodied in the story of soil, which enabled life to make the transition from

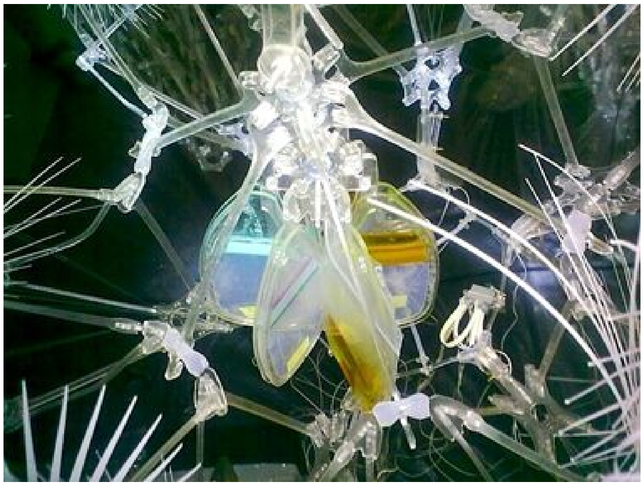


Figure 12: [Photograph, Rachel Armstrong, Hylozoic Ground installation, Venice Architecture Biennale 2010.] Clusters of vertically mounted Liesegang ring plates were introduced as a time-based chemical system in the Hylozoic Ground matrix, like the bark of a tree.

a watery, to a land based environment. The Earth was not 'born' with soil but has acquired its living web of relationships over the millennia (Logan, 2007), which form living bodies on a geological scale. Soils are integrating infrastructures on an architectural scale and attractors of terrestrial life. They enable materials to be transformed in the free flow of elemental systems through their bodies, such as air, water, heat and matter. This innate complexity may have been key to biogenesis, which has been recently suggested in work with clays that act as biotic catalysts, such as montmorillonite (Hanczyc et al., 2003). Indeed, William Bryant Logan proposes 'the clay code' is "*more complex than either the genetic code or human language*" (Logan, 2007, p127).

Yet soils are more than just their chemical ingredients, they are a highly structured spatial system and processing platform that promotes lifelike and living systems. They possess a reticular network of channels and spaces through which elemental systems can readily flow. The journey and topology of these labyrinthine spaces plays an important role in provoking complexity in the matter that passes through – and even becomes assimilated by – the soil body. This means that within a soil structure, matter is processed differentially and substrates are subject to different conditions depending on their position within the soil matrix. While we take many of these operations for granted in a terrestrial context, their complex spatial and chemical relationships are essential for generating the sustained metabolic networks that form our ecologies. Notably, the matter that passes into a soil system also becomes an integral part of the soil body – not just structurally but also physiologically.

The active spatial programming of soils through physiological processes was documented by Charles Darwin, who observed that earthworms were responsible for the movement of large stones into the earth. In this context, worms acted as a kind of local 3D printer system. They obtained the printing material by removing dirt from underneath the rocks and then re-depositing it on to the surface as casts. The stones therefore sunk into the ground faster than gravity alone (Darwin, 2007).

Conjecturally, soils may be considered as a production platform for dissipative structures, elemental matrices and natural computing tactics, which collectively contribute to a process of dynamic complexification. The strategic manipulation of these relationships may increase the probable emergence of lifelike systems – but also sustain and propagate them once they have arisen. Indeed, the production of 'artificial' soils may be regarded as an appropriate area for scientific research in the origins of life portfolio – and may provide new insights into generating environments in which the persistence of lifelike events becomes increasingly likely.

Conclusion

While origins of life sciences have generally been regarded as blue-sky sciences, the insights developed in the late 20th century have enabled designers to work with lifelike properties and non-equilibrium material systems. The origins of life sciences may help us deal with more complex and dynamic physical realm, where the process of discovery, language generations and cultural contingency can greatly enrich developing practices. Together, design and the origins of life sciences may generate new technological platforms that can produce a new portfolio of solutions – where 'life' itself may not be regarded as the only end point of research but is part of a whole new range of technical systems that increase environmental liveliness in an ecological era.

References

Adamatzky, A., Bull, L., Costello, B. D. L., Stepney, S., and Teuscher, C., editors (2007). *Unconventional Computing 2007, Bristol, UK, July 2007*. Luniver Press.

Adamatzky, A., De Lacy Costello, B., and Asai, T. (2005). *Reaction-Diffusion computers*. Elsevier Science.

Armstrong, R. (2015). *Vibrant Architecture: Material Realm as a Codesigner of Living Spaces*. De Gruyter.

Armstrong, R. and Hanczyc, M. (2013). Bütschli dynamic droplet system. *Artificial Life Journal*, 19(3-4):331–346.

Berger, P., Adelman, N. B., Beckman, K. J., Campbell, D. J., Ellis, A. B., and Lisensky, G. C. (1999). Preparation and properties of an aqueous ferrofluid. *Journal of Chemical Education*, 76:943–948.

Darwin, C. (2007). *The Formation of Vegetable Mould Through the Action of Worms (reprint)*. The Echo Library.

Graziani, K. R., Hudson, J. L., and Schmitz, R. A. (1976). The Belousov-Zhabotinskii reaction in a continuous flow reactor. *The Chemical Engineering Journal*, 12(1):9–21.

Hanczyc, M. M., Fujikawa, S. M., and Szostak, J. W. (2003). Experimental models of primitive cellular compartments: encapsulation, growth and division. *Science*, 302:618–622.

Hight, C. (2007). *Architectural principles in the age of cybernetics*. Routledge.

Ikegami, T. and Hanczyc, M. (2009). Search for a first cell under the maximalism design principle. *Technoetic Arts*, 7(2):153–164.

Latham, W. (1988). Conquest of form: Computer art by william latham. Exhibition at Arnolfini Gallery, Bristol.

Lee, M. (2011). *Oceanic Ontology and Problematic Thought*. NOOK Book/Barnes and Noble.

Liesegang, R. E. (1896). Ueber einige Eigenschaften von Gallerten. *Naturwissenschaftliche Wochenschrift*, 11(30):353–362.

Logan, W. B. (2007). *Dirt: The Ecstatic Skin of the Earth*. W. W. Norton & Company.

Pfeifer, R. and Iida, F. (2005). Morphological computation: Connecting body, brain and environment. *Japanese Scientific Monthly*, 58(2):48–54.

Teatini, P., Castelletto, N., Ferronato, M., Gambolati, G., and Tosi, L. (2011). A new hydrogeologic model to predict anthropogenic uplift of Venice. *Water Resources Research*, 47(12):W12507.

Toyota, T., Maru, N., Hanczyc, M. M., Ikegami, T., and Sugawara, T. (2009). Self-propelled oil droplets consuming “fuel” surfactant. *Journal of the American Chemical Society*, 131(14):5012–5013.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 237(641):37–72.

Wakefield, J. (2013). TED 2013: 4d printed object ‘make themselves’. BBC news technology. http://www.bbc.co.uk/news/technology-21614176 [Accessed 1 March 2015].

Wolchover, N. (2014). A new physics theory of life. *Quanta*. https://www.quantamagazine.org/20140122-a-new-physics-theory-of-life/. [Accessed 10 February 2015].

Hybrid Forms (2015)

Andy Lomas

http://www.andylomas.com/

Artist Statement

Hybrid Forms is a new work created for ECAL 2015, and represents the latest stage in a series of work called Cellular Forms. Inspired by Alan Turing, Ernst Haeckel and D'Arcy Thompson, these works use simplified biological models of morphogenesis to explore the generation of complex three dimensional structures.

Each form starts with a small initial ball of cells which is incrementally developed over time, adding iterative layers of complexity to the structure. The aim is to create forms emergently from the interactions between individual cells, exploring generic similarities between many different shapes in nature rather than emulating any particular organism. The process reveals universal archetypal forms that can come from growth-like processes rather than top-down externally engineered design.

Cell division is controlled by accumulated nutrient levels. When the level in a cell exceeds a given threshold the cell divides, and various parameters control how both the parent and daughter cells re-connect to their immediate neighbours. Rules can also be adjusted for how nutrient is created, such as by being randomly uniformly created by each cell, or by incident light rays creating nutrient in cells hit by photons. Nutrient can also be allowed to flow to adjacent cells. The simulation process is repeated over thousands of iterations and millions of particles, with typical final structures having over fifty million cells.

A number of internal forces affect the structures, including linear and torsion springs between connected cells. Additional forces repel cells that are in close proximity but are not directly connected. This creates tensions within the structures that induce them to change shape dynamically, with surfaces naturally folding into complex organic forms.

In these 'hybrid' structures diversity is introduced by using two cell types, with different parameters controlling things such as rate of growth and the forces that mediate interaction between the cells. This causes many different types of complex structures to form as regions with different cell types interact with each other.

A wide set of variations arise from small changes to the rules governing the systems, with selection of forms based on aesthetic considerations rather than optimizing a conventional fitness function. All resultant motion as well as shape is genuinely emergent, since the simulation rules only dictate interactions between adjacent cells in the structures.

Medium

Four high definition video streams with accompanying audio. 16 minutes.

Biography

Andy Lomas is a digital artist and Emmy award winning supervisor of computer generated effects. Cellular Forms is the latest part of Morphogenetic Creations: a series of work which explores how complex organic structures, such as those seen in nature, can be the emergent generative products of growth processes.

In 2014 Cellular Forms won The Lumen Prize Gold Award, as well as the Best Artwork Award from the A-Eye exhibition at AISB-50, and an Honorary Mention from the jury at the Ars Electronica Festival.

He has had work exhibited in over 50 joint and solo exhibitions, including SIGGRAPH, the Japan Media Arts Festival, the Ars Electronica Festival, the Los Angeles Center for Digital Art and the Centro Andaluz de Arte Contemporaneo. He has work in the D'Arcy Thompson Art Fund Collection, and was selected by Saatchi Online to contribute to a special exhibition in the Zoo Art Fair at the Royal Academy of Arts.

His production credits include Walking With Dinosaurs, Matrix: Revolutions, Matrix: Reloaded, Over the Hedge, The Tale of Despereaux, Avatar, and he received Emmys for his work on The Odyssey (1997) and Alice in Wonderland (1999).

Do Endothelial Cells Dream of Eclectic Shape?

Katie Bentley

Computational Biology Laboratory
Center for Vascular Biology Research, BIDMC, Harvard Medical School
99 Brookline Avenue, Boston, MA 02215, USA
E-mail: kbentley@bidmc.harvard.edu

Abstract

Endothelial cells (ECs) line blood vessels and exhibit dramatic plasticity and diversity of form/behavior at the individual and collective cell levels. ECs coordinate in space and time to extend new blood vessel networks into tissues low in oxygen. Using examples from our integrated research program, we will descrive how the Artificial Life perspective and approaches can be utilized to drive entirely new experimental biology based discoveries by capitalizing on the emergent behavior, predictive capacity and testable nature of agent-based models in close combination with *in vitro* and *in vivo* experiments.

Introduction

We can learn a great deal about real endothelial cells (ECs) by watching the interaction of simulated ECs in the "Virtual Lab" as they collectively generate new and unexpected tissue-level dynamics. This is detailed in our recent perspectives article (Bentley et al 2014a) and can be illustrated by drawing parallels with the thought-provoking robotic humans (androids) in Philip K. Dick's novel "Do Androids Dream of Electric Sheep?". In both the book and the film adaptation (Ridley Scott's "Bladerunner") android behavior serves as a mirror to view, question and understand human behavior. Taking the Electric sheep/Bladerunner analogy further, we aspire to study ''rogue simulant cells", instantiated with mutations and/or let loose within untested pathological environments. Their unexpected aberrant behavior, we will show, can provide solid predictions, for new *in vitro* and *in vivo* experiments, providing insight into mechanisms behind maladapted behavior of real ECs and therapeutic strategies.

Results

Understanding how, when, and why individual ECs coordinate their decisions to change shape, move and interact in order to grow functional blood vessel networks ("angiogenesis") in relation to the myriad of dynamic environmental signals around them, is key to understanding normal and pathological blood vessel behavior. This is a complex, spatial and temporal problem, however. Each cell's individual autonomy in determining its own, time-variable behavior is not easy to extrapolate from everyday experimental techniques, which often provide instead averaged or static, population-level data. Here, we will show, using several examples from our published and unpublished research, that agent-based models of endothelial cell dyamics integrated with *in vitro* and *in vivo* experimentation, can lead to new mechanistic insight into normal and abnormal cancerous/retinopathy blood vessel growth. We explictly consider the role of individual EC embodiment, active perception, heterogeneous vs homogeneous collective dynamics, pattern formation processes and counter-intuitive emergence from feedback in controller networks (Bentley et al 2008, 2014b, 2014c).

We no longer see the big challenge ahead as whether or not simulations that capture aspects of cellular systems can be built. This has been achieved. The challenge we set ourselves is: can we gain novel, experimentally relevant insight with simulations, that when tested will generate new insights for the experimental biology community as well as for theoretical biologists and the ALife community. We will discuss the practical approaches and guidelines we employ to meet this challenge (detailed in Bentley et al 2014a).

Conclusions

Alife is a perfect mindset and perspective with which to understand spatial autonomous adaptive behavior of cells in biological systems such as the vasculature. If we take care to be rigorous in how we callibrate our models to biological data and make clear experimentally testable predictions, we can drive experimental biology forward. Learning from the insightful, but segregated Androids in Philip K Dick's novel, if we can overcome our cultural differences and integrate our knowledge better between artifical and natural systems, we can together tackle the full and complex array of mechanisms driving coordinated cell behaviors in living systems.

References

Bentley, K., Philippides, A., Ravasv Regan, E. (2014a). "Do Endothelial Cells Dream of Eclectic Shape?" *Developmental Cell* **29(2):**146-158.

Bentley, K., Franco, C. A…Gerhardt, H (2014b). "The role of differential VE-cadherin dynamics in cell rearrangement during angiogenesis." *Nature Cell Biology*, **16**:309-32.

Bentley, K, Harrington K., Ravasv Regan, E. (2014c). "Can active perception generate bistability? Heterogeneous collective dynamics and vascular patterning". *Proc. ALife 14*. 328-335.

Bentley, K., Gerhardt, H., and Bates, P. A (2008). "Agent-based simulation of notch-mediated tip cell selection in angiogenic sprout initialisation". *J Theor Biol*, **250**:25-36.

Physarum Polycephalum changes polyaniline properties

Alice Dimonte[1], Tatiana Berzina[1] and Victor Erokhin[1]

[1] CNR-IMEM (National Council of the Researches – Institute of Materials for Electronics and Magnetism), Parco Area delle Scienze 37A, 43124, Parma, Italy

alice.dimonte@imem.cnr.it

Abstract

Physarum polycephalum slime mould can modify polyaniline (PANI) features due to its internal activity. We created networks with different conductivity made by the slime mould on PANI substrates. Thus, *Physarum's* growth results in changing the conductivity state of PANI layers, providing negative and positive patterning of the samples. A spectrophotometric scanner is here exploited to investigate and characterize the effects coming out from the interaction between *Physarum polycephalum* and PANI. The latter is an electro-chromic polymer that vary its colour and conductive properties according to its redox state.

Introduction

Physarum polycephalum slime mould is attracting interest of scientists, not only biologists, because of its incredible features and adapting capabilities. Indeed, since many years, it has been the object of studies in the field of unconventional computing, (Gale et al. 2013), networks modelling and development (Nakagaki et al. 2004), biorobotics (Mayne et al. 2013), and biochemistry (Romeo et al. 2015). *Physarum polycephalum* slime mould is a single cell organism with many nuclei dispersed in the cytoplasm. It belongs to the family of Myxomycetes, Physarales species, taxonomically classified as Protozoas or, simply, slime mould (Ratzel et al. 2013). In nature it feeds on bacteria and decaying organic materials and needs darkness, humidity and a temperature around 25-27°C. Therefore, the abovementioned characteristics have to be considered in order to properly culture a colony in a laboratory. *Physarum polycephalum* can be, thus, cultivated on agar gel or wet towels kept in a dark and humid chamber and fed with oat-flakes. Moreover, to ensure its safety, the colony has to be periodically replanted to a new fresh substrate (agar or towels). *Physarum polycephalum* life is characterized by different phases (Stephenson et al. 2000): sporulation, sclerotium and plasmodium. The first is the reproduction phase, exploited by meiosis and fructification; the second is the dormant phase, a sort of hibernation during which the mould protect itself from not proper environmental conditions. The latter, plasmodium, is the vegetative form of slime mould; during this phase the organism is more active and moves searching for food. It appears as a yellowish multinuclear mass of protoplasm, a single cell with a myriad of nuclei, able to regenerate autonomously, even cutting a part away. Therefore, it has not a fixed mass, but it continuously changes its shape as a function of the spatial food distribution; indeed, it has been also demonstrated its capability of choosing not only the closest but also the most nutritional food sources from a buffet (Latty et al. 2009). Figure 1 shows an example of the abovementioned feature. Its foraging behaviour consists in the formation of optimized networks (Bonifaci et al. 2012) of protoplasmic tubes branching towards nutrients. However, in the case of food abundance it tends to stasis creating "pancakes-like" structures that enwrapped the whole food allowing phagocytosis (Golderer et al. 2001). These are astonishing features for a simple amoeba. In addition, as its name suggest, it produces a certain amount of slime, an extracellular polysaccharide (Simon et al. 1970) working as a sort of "external brain". The slime allows *Physarum polycephalum* to remember already trodden paths, in order to not retrace them when seeking for food or other attractants (Nakagaki et al. 2001). The protoplasmic veins of plasmodium's network consist in a two phases sol-gel medium. The external, more rigid, gel is the ectoplasm and contains the sol, more fluid endoplasm (Wohlfarth-Bottermann, 1974) which is transported by means of proteins present in the ectoplasm, the actomiosins. The latter, by contracting, generates high-pressure gradients that push the endoplasm forward allowing locomotion of the whole organism, in the so called shuttle streaming mechanism (Matsumoto et al. 2008). The way by which *Physarum* feels attractors and food in a certain direction, as well as the individuation of the forces involved in the locomotion of this organism, are still unknown. In this interesting scenario, we exploited *Physarum*'s capability of creating networks (Shirikawa and Gunji, 2007) to pattern polyaniline (PANI) samples.

PANI is a redox electro-chromic polymer widely studied, also by authors for organic memristive device realizations (Erokhin and Fontana, 2011). Moreover, a spectrophotometer device has also been used to get evidence of the modifications induced by the mould on the PANI samples. In this work, we found a method by which it is possible to characterize *Physarum's* networks by electrical and spectrophotometric measurements,. The latter, built by the mould onto PANI substrate, were transferred, as a lithographic process, on PANI itself.

Therefore, we finally project *Physarum* done networks on the PANI layer.

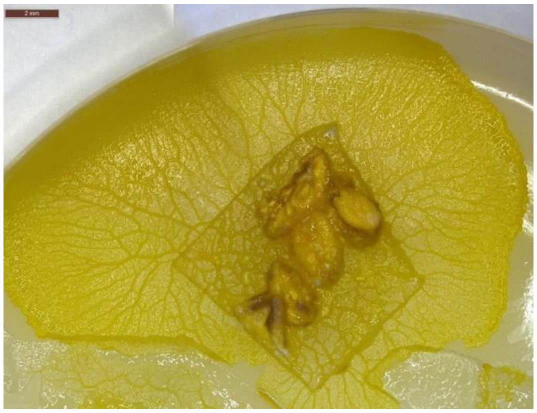


Figure 1: Optical microscope image of Physarum polycephalum on Agar nutrient gel, in this case it is clearly visible that the mould is creating a sort of "pancake-like" structure growing in all directions homogeneously. Thus demonstrating its shape is a function of spatial food distribution.

Materials and methods

Physarum culture

Physarum polycephalum slime mould, as mentioned in the introduction section, was cultured in dark and humid chamber at room temperature. The colony was maintained in Petri dishes with agar 1.5% non nutrient gel, fed with oat-flakes and periodically replanted to a new fresh agar.

Spectrophotometer

The device (Dimonte et al. 2015), a spectrophotometric scanner, is generally exploited in the art and restoration fields, being able to appreciate chromatic differences with high spatial resolution between points. The instrument produces an image consisting in an ordered collection of the spectral reflectance factors of each pixel.

The spectrophotometer here used is composed of a transmission spectrometer (Imspector V8 manufactured by Specim, Finland) designed for a 2/3 inch CCD sensor equipped with a 25 µm entrance slit and covering the 400÷780 nm spectral range with a spectral resolution of about 2 nm. The spectrometer is coupled to a monochrome 2/3 inch CCD matrix chill digital camera (Hamamatsu C4742-12bit, 1280×1024 pixels, 9 f/sec) while a collecting lens (Computar TEC-M55 designed for a 2/3 inch sensor) focalizes the painting on the plane of the entrance slit. The illumination is obtained by means of two 150 W halogen lamps whose light is filtered preventing the illumination of the painting. The digital camera is interfaced to a PC by means of a 12 bit frame grabber (Mutech MV1000). A software program drives the scanner, acquires data of a strip of the scene and allows to save its image as a spectral image. The program is implemented to reproduce the colours on a calibrated CRT monitor. The light is dispersed by the spectrometer and focalized in the plane, containing the sensor of the camera. The spectrometer has a 1:1 image magnification then, the image of the input is focalized on the pixel rows of the sensor, while its position along the vertical axis of the sensor, depends on the light wavelength. White light in the range 400÷780 nm enterig into the device fills the whole sensor. The acquisition of one frame of the digital camera can be thought as the acquisition of the reflectance spectra of each pixels of the strip of the painting.

Polyaniline samples

Polyaniline is a redox electrochromic polymer, discovered in 1985 (Kang et al. 1998). Its main feature is represented by a high conductivity difference between the conductive, green, oxidized state and the insulating, blue reduced one (Bredas et al. 1985). The transition depends on the doping degree.It is reversible and can be achieved by chemical treatment (i.e. water or basic solutions for reducing and acid chloride for oxidazing) or by proper voltage application (Berzina et al. 2007). Emeraldine base polyaniline was purchased from Sigma (Mn ca. 100,000). The deposition of the active PANI layer was carried out with a KSV 5000 LB trough, using a Langmuir–Schaefer technique. Pure water, purified with a Milli-Q system, with a resistivity of 18.2 MΩcm, serves as the subphase (Dimonte$_2$ et al. 2014). Polyaniline powder was dissolved in l-methyl-2-pyrrolidonesNMPd and carefully filtered. The real concentration of the solution was determined and then NMP was added to achieve the final concentration of 0.2 mg/ml.

Oxygen-Containing Plasma Treatment

The treatment has been performed with a Plasma Matrix bdiscom machine with a 10 min exposure at 99 W. This allows us to remove PANI from the areas of the samples not covered by mould's network.

Optical measurements

The samples have been characterized by means of an optical microscope Leica D300. In addition, it was also fundamental, at the final state of the work, to make proper contacts of the PANI networks by means of micro-manipulating tips.

Electrical characterizations

The electrical conductivity of the PANI networks was recorded by means of a Keithley 2400 SourceMeters and two micromanipulators ending with tungsten tips.

Experiments and Results

Polyaniline samples were prepared by depositing 40 molecular layers of PANI on glass substrates by Langmuir-Shaefer technique (see material and methods); afterward the layer was doped by HCl treatment.

We put 2 µl blob of *Physarum polycephalum* in the centre of a glass with deposited PANI film and attractors (oat flakes) at the external boundaries, to stimulate the mould in exploring the sample. Therefore, *Physarum polycephalum*, spanning towards the food sources in 8-12 hours, created networks on the polymer substrate as it is shown in Figure 2a).

The sample was then transferred to open air and light conditions to let *Physarum* enter in the sclerotic phase and, consequently, maintain unvaried the designed network. Subsequently, we applied Oxygen plasma to remove PANI in all samples areas not covered by the mould.

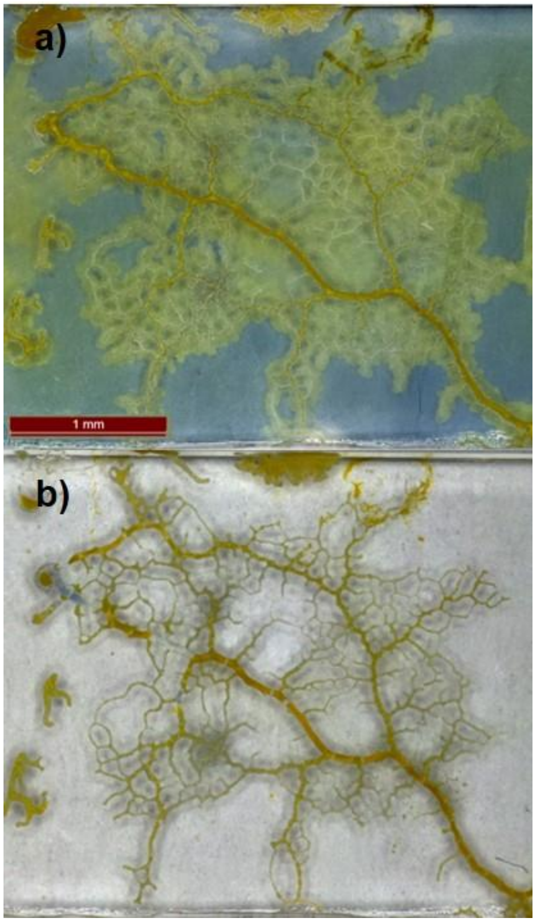


Figure 2 Optical microscope photograph of a network created by *Physarum polycephalum* on PANI sample before (part a) and after (part b) oxygen plasma treatment.

The result is shown in Figure 2b). The reported picture allows to conclude that the performed treatment did not destroy the network and the sclerotized mould stays attached to the surface without obvious damages. Successively, the mould was removed from the support surface by dipping the sample in water and than washing it gently. As expected, underneath the sclerotium we have found the PANI network. The PANI was doped to transfer it into conductive state by acid treatment (see materials and methods), we then contacted the channels of the network by means of micro-manipulating tips connected to a measurement station and verified that the most of channels were rather conductive with a calculated resistance in the range of 24-70 MOhm depending on the tested areas. Figure 3 shows the analysis scheme developed on the typical patterned samples. In particular it is clearly visible the green polyaniline network designed by *Physarum*.

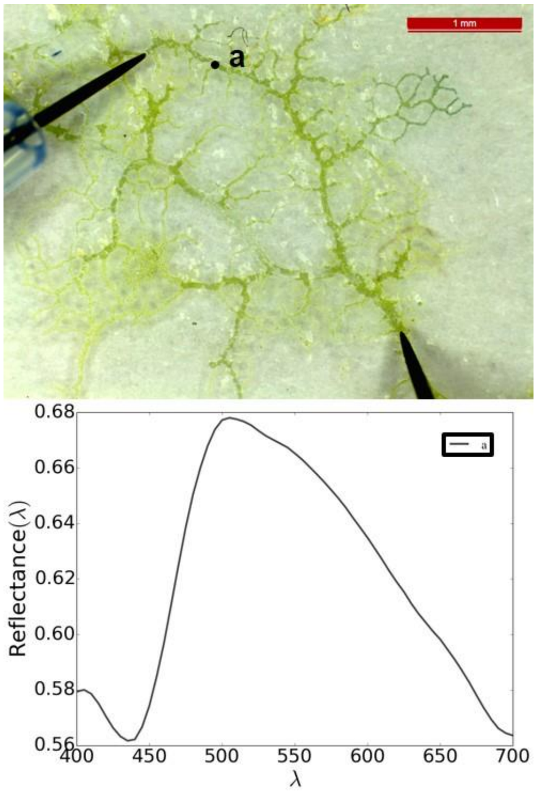


Figure 3: The part above shows an optical microscope photograph of an analyzed sample after Oxygen Plasma treatment and removal of the mold. The 2 black tips are the contact points for electrical characterization and the resulting value is 24 MΩ. In the part below, the spectrum refers to the point "a" signed in the image. Since the picture is a collection of spectra, it is possible to obtain spectra from lines or areas by selecting them directly on the image.

The latter can be considered as a type of lithography. Thw pattern, created during the slime mold growth can be transferred on the PANI layer, developed by the oxygen plasma and realizing the conductivity patterning after the acid treatment doping. Moreover, micromanipulating tips allow contacting the channels, one at a time, and checking the conductivity in desairable channels. The point marked on the image corresponds to the spectrum below. Indeed, after electrical characterizations, samples were measured with spectrophotometer and data have been elaborated considering pure polyaniline film as a reference.

Conclusions

The study here presented shows interesting results and opens many possible developments. Considering the non-invasiveness of the spectral analysis method (no influence on the conductivity variations), it will be possible to perform measurements in real time, characterizing the network during the motion of *Physarum polycephalum*. Therefore, we connected the growth of the slime mould with a variation of optical and electrical properties of the sub-layer (Dimonte et al. 2014). Here, putting a step forward, we demonstrated not only that *Physarum polycephalum* induces redox reactions, but also we exploited Physarum polycephalum's network as a lithographic mask. Therefore, we found and elaborated a technique by which it is possible to design conductive channels of polyaniline. Moreover, we applied the spectral imaging method for characterizing the patterned polyaniline surfaces formed during the *Physarum polycephalum* growth. Considering such kind of images, it is possible to appreciate chromatic differences with high spatial resolution between space-separated points. This is an important feature for the characterization of the grown networks of *Physarum polycephalum*. Therefore, the spectrophotometric scanner is here exploited in a non-conventional context, but it helps us to visualize the interactions between the growing mould and polyaniline layer.

Acknowledgments. This work is supported by the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the Collaborative project PhyChip, grant agreement number 316366.

References

Berzina, T., Erokhin, V. and Fontana, M. P. (2007) Spectroscopicinvestigation of an electrochemically controlled conducting polymer-solid electrolyte junction. *J. Appl. Phys.*, 101: 024501.

Bonifaci, V., Mehlhorn, K., Varma, G. (2012). Physarum can compute shortest paths. *Journal of theoretical biology*, 303:121-33.

Brédas, J. L. and Street, G. B. (1985). Polarons, bipolarons and solitons in conducting polymers. *Acc. Chem. Res.*, 18: 309-315.

Dimonte, A., Berzina, T., Cifarelli, A., Chiesi, V., Albertini, F. and Erokhin, V. (2014). Conductivity patterning with Physarum polycephalum: natural growth and deflecting. *Phys. Status Solidi C*, 12:197-201.

Dimonte$_2$, A., Pavesi, M., Berzina, T. and Erokhin, V. (2014). Hysteresis loop and cross-talk of organic memristive devices. *Microelectronics Journal*, 45(11):1396-1400.

Dimonte, A., Fermi, F., Berzina, T., Erokhin, V. (2015) Spectral imaging method for studying Physarum polycephalum growth on polyaniline surface. *Materials Science and Engineering C*, 53:11-14.

Erokhin, V., Berzina, T. and Fontana, M. P. (2005). A polymer based electrochemical device. *J. Appl. Phys.*, 97: 064501.

Erokhin, V., Fontana, M. P. (2011). Thin film electrochemical memristive systems for bio-insoired computation. *J. Computation and Theor. Nanosci* , 8: 313-330

Gale, E., Adamatzky, A. and D. L. Costello, B. (2013). Are slime moulds living memristors? e-print arXiv:1306.3414.

Golderer, G., Werner, E., Leitner, S., Gröbner, P. and Werner-Felmayer, G. (2001). Nitric oxide synthase is induced in sporulation of Physarum polycephalum .*Genes & Dev*.,15: 1299-1309

Kang, E., Neoh, K. and Tan, K. (1998). Polyaniline: a polymer with many intrinsic redox states. *Progress in Polymer Science*, 23: 277–324.

Latty, T. and Beekman, K. (2009). Food quality affects search strategy in the acellular slime mould, Physarum polycephlum. *Behavioural Ecology*, 20: 1160-1164.

Matsumoto, K., Takagi, S. and Nakagaki, T. (2008). Locomotive mechanism of Physarum polycephalum based on spatiotemporal analysis of protoplasmic streaming. *Biophysical Journal*, 94: 2492-2504.

Mayne,R., Patton, D., D. L. Costello, B. Patton, R. and Adamatzky, A. (2013). On loading slime mould Physarum polycephalum with metallic particles. Available at http://eprints.uwe.ac.uk/21054/

Nakagaki, T. (2001). Smart behavior of true slime mold in labyrinth. *Res. Microbiol*., 152: 767-770.

Nakagaki, T., Kobayashi, R., Nishiura, Y. and Ueda, T. (2004). Obtaining multiple separate food sources: behavioural intelligence in the Physarum polycephalum. *Proc. R. Soc. Lond. B*, 271-2305-2310.

Rätzel, V., Ebeling, B., Hoffmann, X., Tesmer, J. and Marwan, W. (2013). Physarum polycephalum mutants in the photocontrol of sporulation display altered patterns in the correlated expression of developmentally regulated genes. *Dev. Growth Differ*., 55(2) 247-259.

Romeo, A., Dimonte, A., Tarabella, G., D'Angelo, P., Erokhin, V. and Iannotta S. (2015). A bio-inspired memory device based on interfacing Physarum polycephalum with an organic semiconductor. *APL Materials*, 3: 014909.

Shirikawa, T. and Gunji, Y. (2007). Emergence of morphological order in the network formation of Physarum polycephalum. *Biophys. Chem*., 128:253-260.

Simon, H. L. and Henney, H. R. (1970). Chemical composition of slime from three species of myxomycetes. *FEBS Letters*, 70: 80-82.

Stephenson, S. L. and Stempen, H. Timber Press (2000). *Myxomycetes: A. Handbook of Slime Molds*. Portland, OR

Wohlfarth-Bottermann, K. E. (1974). Plasmalemma invaginations as characteristic constituents of plasmodia of Physarum polycephalum. *J. Cell Sci*. 16:23-32.

Majority Gates and Circular Computation in Slime Mould

Genaro J. Martínez[1,2], Andrew Adamatzky[1], Richard Mayne[1], Fangyue Chen[3], Qinbin He[4]

[1] Unconventional Computing Centre, University of the West of England, BS16 1QY Bristol, United Kingdom
[2] Escuela Superior de Cómputo, Instituto Politécnico Nacional, México
[3] School of Sciences, Hangzhou Dianzi University, Hangzhou, Zhejiang 310018, P. R. China
[4] Department of Mathematics,Taizhou University, Linhai, Zhejiang 317000, P. R. China

Abstract

Slime mould has been proven to be a fruitful living substrate for implementing a wide range of computing circuits from computational geometry to collision-based logical circuits to robot control. It is apparent, however, that constructing working real-time universal processors from the slime mould is non-trivial task. We explore here similarities between development of slime mould protoplasmic tubes, transportation of cytoplasm inside the tubes and dynamics of propagating patterns in cellular automata. Based on these analogies we will propose computing devices realisable in the living slime mould.

Keywords: unconventional computing, slime mould, mobile localizations, competing patterns, logic gates, tubes, cellular automata.

Preliminaries

Slime mould *Physarum polycephalum* is a syncytial single celled eukaryotic organism that is macroscopic in scale and possesses many thousands of nuclei. Both slime mould behaviour patterns and intracellular processes can be interpreted as expressions of unconventional computation. Achieving motility by the contraction of muscle proteins which drive cytoplasmic flows (cytoplasmic streaming), *P. polycephalum* may navigate between spatially distributed nutrient sources, forming an efficient, interconnected networks of tubular structures (protoplasmic 'tubes') in its wake which comprise the body of the cell.

Recently, a number of relevant and interesting experiments were performed to simulate computations and other slime mould phenomena (Adamatzky, 2010), including logic gates (Adamatzky et al., 2010), spacial logical gates (Schumann and Adamatzky, 2011), optically-coupled logic gates (Mayne and Adamatzky, 2015), hybridised slime mould (Mayne and Adamatzky, 2013), microfluidic logic gates (Adamatzky and Schubert, 2014), oscillators (Adamatzky, 2014), and to a number of diverse Physarum machines (Adamatzky, 2009). Several of them were implemented with specific initial conditions, where chemoattractants were used to control the slime mould propagation and modify topology of its protoplasmic networks.

We will discuss draw analogies between cytoplasm transportation inside the protoplasmic tubes, growth of the tubes and patterns growing in one- and two-dimensional cellular automata and speculate on what kind of computational devices can be mapped from the field of cellular automaton computing to slime mould computing devices.

Cytoplasm, protoplasmic tubes, and logic gates

A number of different logic gates derivations were reported with slime mould including conventional logic gates (Tsuda et al., 2004), ballistic logic gates (Adamatzky, 2010), artificial adders (Jones and Adamatzky, 2010), spatial logic gates (Schumann and Adamatzky, 2011), frequency logic gates (Adamatzky et al., 2014), microfluidic logical gates (Adamatzky and Schubert, 2014) and optically-coupled logic gates (Mayne and Adamatzky, 2015). Typically, the stream of information in these propagations are controlled on some T, Y, X, C, K, H-shaped junctions, or some variant of junctions (some samples are illustrated in Fig. 1). Following this principles, we can explore a kind of computation based in competing patterns that can be extrapolated to slime mould quickly.

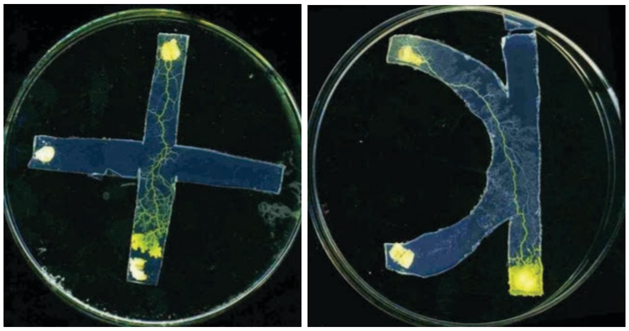


Figure 1: Photographs of exemplar slime mould computing landscapes implementations. (a) X junctions, (b) K junctions.

Cytoplasmic streaming and concurrent propagation of plasmodial tubes along junctions are commonly used in designs of logic gates. In unstimulated, vegetative plasmodia, the organism will propagate towards a food source and cytoplasmic flows oscillate rhythmically through the hydrodynamic core of plasmodial tubes to distribute the absorbed nutrients. Where internalised substances are used to represent data, cytoplasmic streaming may be influenced experimentally by external stimuli (tactile, optical etc.) which will usually reduce local flow in a tube for several minutes. This principle has been used in the fabrication of some of the aforementioned Physarum logic devices. In existing designs of slime mould-based devices the topology of protoplasmic tubes is determined by configurations of attractants and repellents.

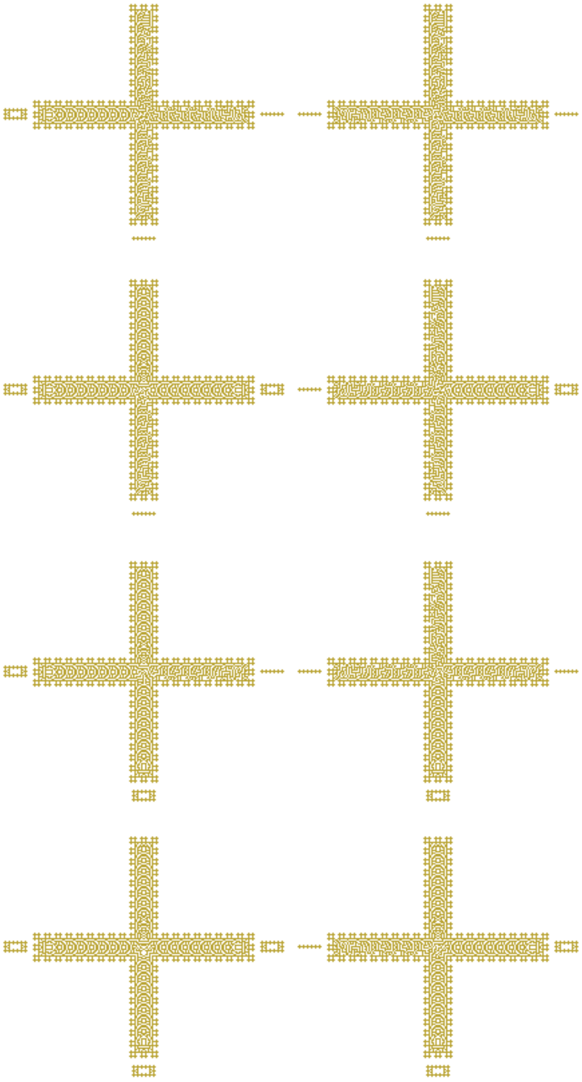


Figure 2: Idealised MAJORITY gate implemented in nonlinear media (computing patterns) in artificial tubes (as networks in slime mould protoplasmic tubes).

Cytoplasmic flow is difficult to influence in any other way, however, as plasmodial behaviour is determined by myriad external and internal factors. In compensation for this, an opportunity to represent binary values can be realised by searching for symmetries in flow patterns through which simple logic gates by pattern propagation may be implemented (Martínez et al., 2008), (Martínez et al., 2010). The representation is simple in the first instance as cytoplasmic streaming must be controlled as a symmetric stream or asymmetric (zero or one). In this way, a number of logical gates can be implemented following a truth table. A salient limitation of this model is, however, that plasmodial tubes are fixed and cannot be removed easily. To solve this situation we can work with cascaded logic gates implemented by analogy with cellular automata, and although cellular automata dynamics seems simple from their own definition, they are powerful tools in simulating complex or chaotic systems. Cellular automata have been demonstrated to be as powerful as Turing machines and there are a number of cellular automata capable of universal computation (Adamatzky, 2002), (Hey, 1998), (Mitchell, 2001), (Martínez et al., 2013), (Wolfram, 1984).

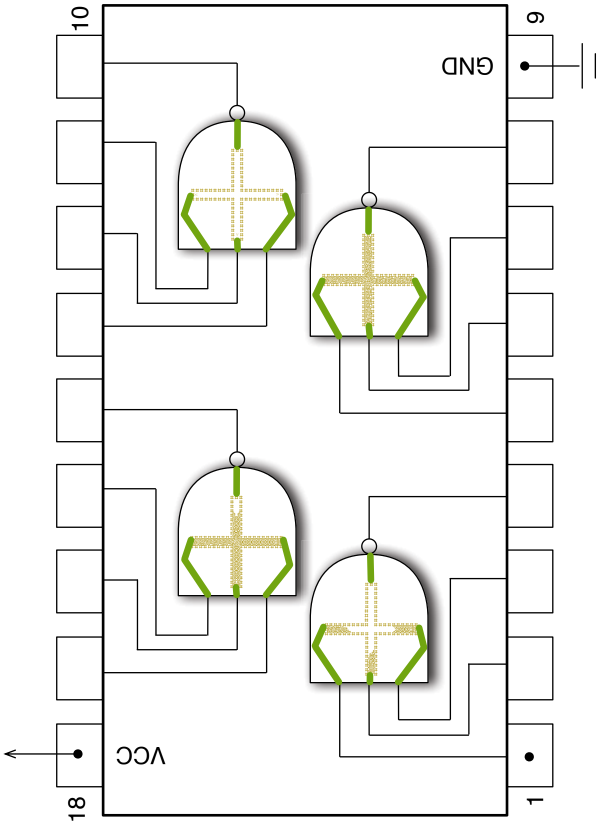
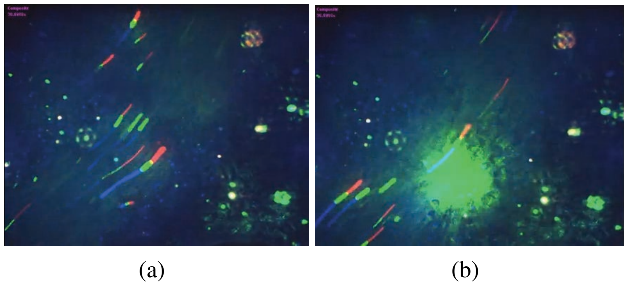


Figure 3: Device 5400/DM5400/DM7400 Quad 2-Input NAND Gates modified to works with MAJORITY gates based competing patterns.

Conventional AND, OR, NOT gates are relatively easy to implement in the Life-like cellular automata (Martínez et al., 2010). However, when implementing non-serial logical gates it is more convenient to work with MAJORITY gates, as they can be extrapolated to slime mould in a more realistic way (these kind of gates are used recurrently in quantum dots cellular automata, see (Gregory et al., 1999)). Figure 2 shows how a MAJORITY gate operates. In this case, these patterns will are encoded within the interior of protoplasmic tubes. We can utilise the scheme to design a modified chip related to 7400 chip from National semiconductor web site. Device 5400/DM5400/DM7400 Quad 2-Input NAND Gates `http://www.icpdf.com/NSC_datasheet/DM5400_pdf_546245/`, but with four MAJORITY and NOT gates instead of four NAND gates, working with three independent inputs per gate on 18 pins. This circuit is designed and illustrated in Figure 3. This way, each MAJORITY gate produces outputs depending of the 3-input values (eight possibilities Fig. 2). Additionally, the NOT gate can be implemented in circuits of competing patterns dynamics, as it was done in *hot ice* computers (Adamatzky, 2009).

Plasmodial tube and circular computation

By loading the plasmodium with a range of nanomaterials (metallic particles, fluorescent latex beads etc.) via vesicular endocytosis or microinjection, plasmodial bioelectrical activity may be altered and collisions between fluorescent particles may be observed, both of which may be used to represent information storage and processing (Mayne et al., 2011), (Mayne and Adamatzky, 2013). Figure 4 shows two snapshots of the particles travelling and colliding with stationary or mobile elements.



(a) (b)

Figure 4: Confocal micrographs showing microscopic fluorescent particle dynamics in the *P. polycephalum* plasmodium (a) Particles traveling diagonally through the endoplasm of a plasmodial tube, and (b) shows the exact instant where an interaction is given. (Images captured from the video `https://www.youtube.com/watch?v=nYLmlFlV4sQ`)

Hypothetically, therefore, a number of plasmodial tubes carrying interacting exogenous particles may be used to design a digital circuit able to perform a computable function. Towards implementing such a system, two difficulties must be overcome: first, to control collisions between individual or groups of particles, and second, to reproduce and synchronise such a reactions periodically. These can be tackled by using our designs of *cellular automata collider* (Martínez et al., 2011) and approaches in *computing with rings* (Martínez et al., 2012).

An evolution space of one-dimensional cellular automata is able to produce particles, gliders, that emerge in complex patterns with a unique identity (period, speed, mass, volume, displacement). In this way, a number of particles are coded as regular expressions that can be manipulated to represent collisions and a number of nano component designs: computable devices, nano-structures, solitons, more complex particles, and beyond.

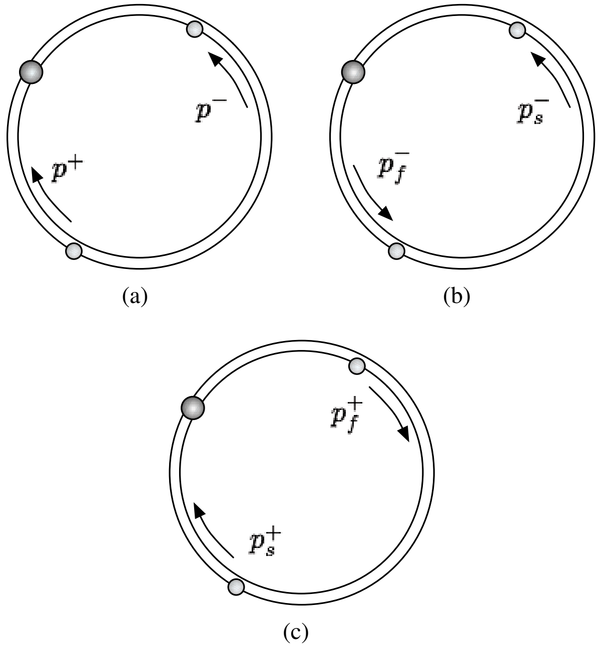
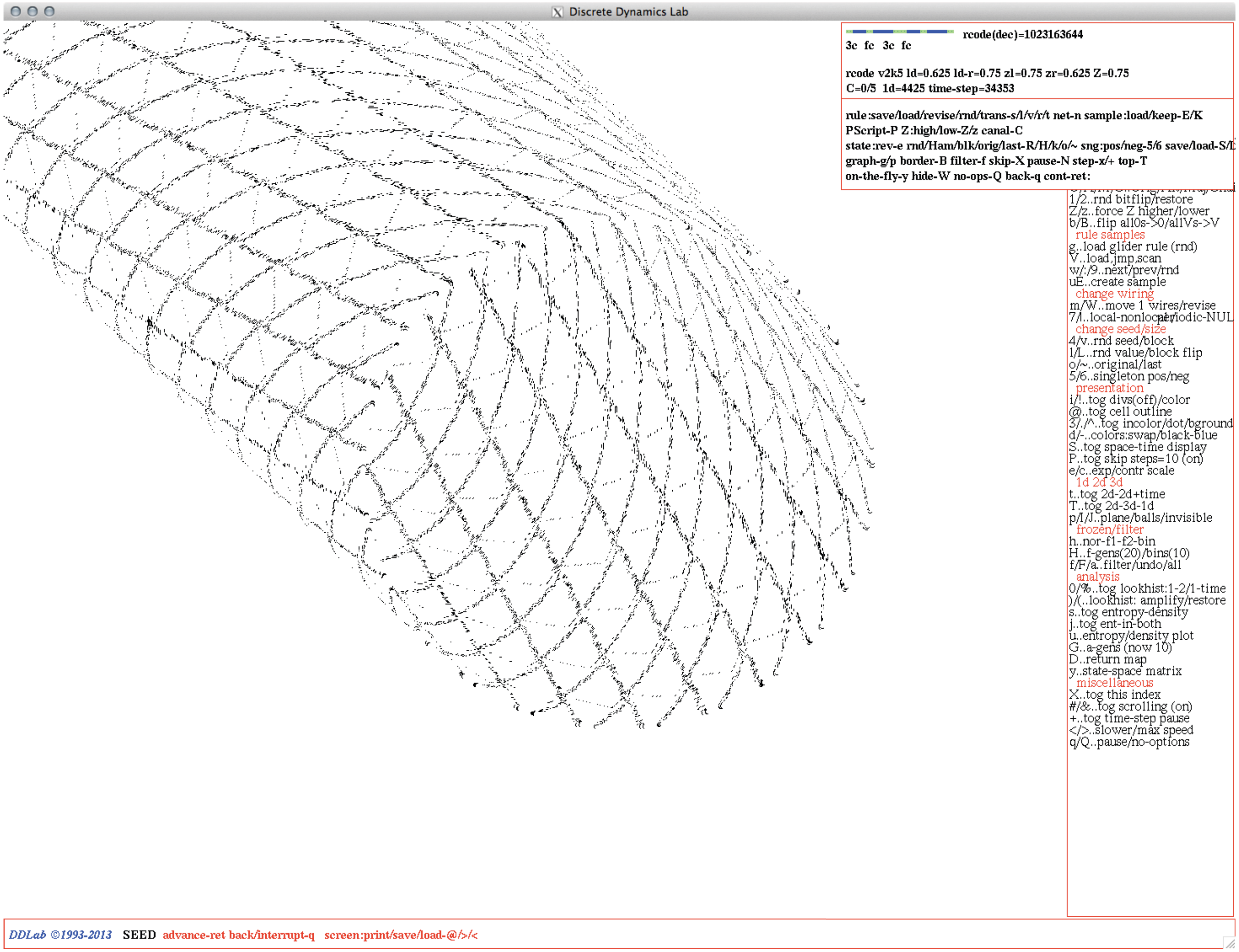


Figure 5: Representation of abstract particles in a one-dimensional cellular automata cyclotron.

Designs for computing these tubes follow a principle based on an unconventional computing representation paradigm (Mills, 2008). In this way, Fredkin and Toffoli have developed a concept of a general-purpose computation based on ballistic interactions between quanta of information that are represented by abstract particles (Fredkin and Toffoli, 2002). The Boolean states of logical variables are represented by balls or atoms, which preserve their identity when they collide with each other. Their 'billiard-ball model' of computation utilises underpinning mechanics of elastically colliding balls and mirrors reflecting the balls' trajectories. Later, Margolus proposed a special class of cellular automata which implement the billiard-ball model. Margolus' partitioned cellular automata exhibited computa-



```
rcode(dec)=1023163644
3c  fc  3c  fc

rcode v2k5 ld=0.625 ld-r=0.75 zl=0.75 zr=0.625 Z=0.75
C=0/5  1d=4425 time-step=34353

rule:save/load/revise/rnd/trans-s/l/v/r/t net-n sample:load/keep-E/K
PScript-P Z:high/low-Z/z canal-C
state:rev-e rnd/Ham/blk/orig/last-R/H/k/o/~ sng:pos/neg-5/6 save/load-S/L
graph-g/p border-B filter-f skip-X pause-N step-x/+ top-T
on-the-fly-y hide-W no-ops-Q back-q cont-ret:

1/2..rnd bitflip/restore
Z/z..force Z higher/lower
b/B..flip all0s->0/allVs->V
  rule samples
g..load glider rule (rnd)
V..load,jmp,scan
w/:/9..next/prev/rnd
uE..create sample
  change wiring
m/W..move 1 wires/revise
7/l..local-nonlocal/periodic-NUL
  change seed/size
4/v..rnd seed/block
l/L..rnd value/block flip
o/~..original/last
5/6..singleton pos/neg
  presentation
i/!..tog divs(off)/color
@..tog cell outline
3/./^..tog incolor/dot/bground
d/-..colors:swap/black-blue
S..tog space-time display
P..tog skip steps=10 (on)
e/c..exp/contr scale
  1d 2d 3d
t..tog 2d-2d+time
T..tog 2d-3d-1d
p/I/J..plane/balls/invisible
  frozen/filter
h..nor-f1-f2-bin
H..f-gens(20)/bins(10)
f/F/a..filter/undo/all
  analysis
0/%..tog lookhist:1-2/1-time
)/(..lookhist: amplify/restore
s..tog entropy-density
j..tog ent-in-both
u..entropy/density plot
G..a-gens (now 10)
D..return map
y..state-space matrix
  miscellaneous
X..tog this index
#/&..tog scrolling (on)
+..tog time-step pause
</>..slower/max speed
q/Q..pause/no-options

DDLab ©1993-2013   SEED   advance-ret back/interrupt-q   screen:print/save/load-@/>/<
```

Figure 6: Cellular automata tube computing a simple oscillator simulating a structure such a carbon or graphene nanotube based particle collisions. Simulation done with DDLab (Dynamics Discrete Lab, `http://www.ddlab.org` (Wuensche, 2011).

tional universality because they simulated Fredkin gate via collision of soft spheres (Margolus, 2002).

Basic functions with two input arguments u and v can be expressed via collision between two localizations as shown in Fig. 6:

1. $f(u, v) = c$, fusion.

2. $f(u, v) = u + v$, interaction and subsequent change of state.

3. $f_i(u, v) \mapsto (u, v)$ identity, solitonic collision.

4. $f_r(u, v) \mapsto (v, u)$ reflection, elastic collision.

To map Toffoli's supercollider (Toffoli, 2002) onto a one-dimensional cellular automata we use the notion of an idealised particle $p \in \Sigma^+$ (without energy or potential energy). The particle p is represented by a binary string (regular expressions) of cell states (Martínez et al., 2011).

Figure 5 shows two typical scenarios where particles p_f and p_s travel in a cellular automata cyclotron. The first scenario (Fig. 5a) shows two particles travelling in opposite directions which then collide. Their collision site is shown by a dark circle in (Fig. 5a). The second scenario demonstrates a typical beam routing where a fast particle p_f eventually catches up with a slow particle p_s at a collision site, which is is typically presented in particles emerging in protoplasmic tubes (Fig. 5b). If the particles collide like solitons (Jakubowski et al., 2001), then the faster particle p_f simply overtakes the slower particle p_s and continues its motion (Fig. 5c).

Following these basic principles, we can explore how microscopic fluorescent particles in *P. polycephalum* can be controlled and synchronised to manipulate computable devices in the large space of tubes/cyclotrons. We transduce the fluorescent particles as a set of strings and organise them in a set of valid reactions to convert a small database of these collisions to an algorithm.

The aim of such representation is to develop an automatic process to construct nano-assembly devices for unconventional computing, derived from a set of synchronised collisions between multiple particles. Such an automation is based on programming regular expressions and finite state machines on circular mechanism.

With regards to a complex one-dimensional cellular automaton supporting abstract particles emerging in its evolution space, we can codify a number of particles to synchronise multiple collisions and to simulate a basic oscillator. Figure 6 shows particles travelling in opposite directions: after of each collision between the particles two new particles emerge but they eventually collide and are transformed into the original two particles. These reactions are spatially synchronised and yield a basic oscillator projected on several copies. This pattern also can be seen as a nano-structure constructed from collisions of mobile and stationary localizations in cellular automata and excitable media. This simulation starts with an initial condition of 4,425 cells codifying 25 particles evolving during 34,353 steps. The history of these collisions need 152,012,025 cells.

Cyclotrons and rings are powerful tools to implement sophisticated algorithms and complex codifications from other designers (Cook, 2004), (Wolfram, 2002), (Martínez et al., 2011). It is possible to design and codify complex equivalent Turing machines on these abstract colliders (following strong theories about circular Turing machines, circular Post machines, and cyclic tag systems) (Arbib, 1969), (Kudlek and Rogozhin, 2001), (Martínez et al., 2011).

Conclusion

Slime mould *P. polychepalum* is a powerful living computing substrate. When presented with data encoded in the configurations of attractants and repellents the slime mould works as an efficient specialised processor capable of approximating Voronoi diagrams, concave hull and shortest paths, and may solve many other computational problems. So far no general purpose computer chip has been developed with slime. This is due to difficulties in interfacing the protoplasmic tubes with conventional electronic components, due in part to the low conductivity of the protoplasmic tubes and instability of the tubes' topology. We have here provided a brief insight on how universal computing devices can be implemented with the slime mould's protoplasmic tubes. These analogies were drawn between competitive peristaltic contractions of the tube (cytoplasmic streaming) and a majority gate, where patterns growing inside the gate's channels compete for free space with each other, and cellular automata supercolliders, where gliders represent voltage solitons (Tuszyński et al., 2004) transported along actin filaments of protoplasmic tubes and the computation is implemented via collisions between the solitons. Hopefully, these abstract models will inspire us towards novel design of Physarum chips.

References

Adamatzky, A. (ed.) (2002) *Collision-Based Computing*, Springer.

Adamatzky, A. (2009) Hot ice computer, *Physics Letters A*, 374(2):264–271.

Adamatzky, A. (2010) *Physarum Machines: Computers from Slime Mould*, World Scientific Series on Nonlinear Science, Series A.

Adamatzky, A. (2010) Slime mould logical gates: exploring ballistic approach, arXiv:1005.2301 [nlin.PS].

Adamatzky, A. (2014) Slime mould electronic oscillators, *Microelectronic Engineering Volume*, 124(25):58–65.

Arbib, M. A. (1969) *Theories of Abstract Automata*, Prentice-Hall Series in Automatic Computation.

Adamatzky, A. and Schubert, T. (2014) Slime mould microfluidic logical gates, *Materials Today*, 17(2):86–91.

Adamatzky, A., Whiting, J., and Costello, B. L. (2014) Slime mould logic gates based on frequency changes of electrical potential oscillation, *Biosystems*, 124:21–25.

Cook, M. (2004) Universality in Elementary Cellular Automata, *Complex Systems*, 15(1):1–40.

Fredkin, E. and Toffoli, T. (2002) Design Principles for Achieving High-Performance Submicron Digital Technologies, In: *Collision-Based Computing*, A. Adamatzky (Ed.), Springer, chapter 2, 27–46.

Hey, A. J. G. (1998) *Feynman and computation: exploring the limits of computers*, Perseus Books.

Jones, J. and Adamatzky, A. (2010) Towards Physarum binary adders, *BioSystems*, 101:51–58.

Jakubowski, M. H., Steiglitz, K., and Squier, R. (2001) Computing with Solitons: A Review and Prospectus, *Multiple-Valued Logic*, 6(5-6). In: *Collision-Based Computing*, A. Adamatzky (Ed.), Springer, chapter 10, 277–298.

Kudlek, M. and Rogozhin, Y. (2001) New Small Universal Post Machine, *Lecture Notes in Computer Science*, 2138:217–227.

Mayne, R., Patton, D., de Lacy Costello, B., Adamatzky, A. and Patton, R. (2011) On the internalisation, intraplasmodial carriage and excretion of metallic nanoparticles in the slime mould, *Physarum polycephalum*, *International Journal of Nanotechnology and Molecular Computation*, 3(3):1–14.

Mayne, R. and Adamatzky, A. (2013) Towards Hybrid Nanostructure-Slime Mould Devices, *Nano LIFE*, 4:1450007 .

Mayne, R. and Adamatzky, A. (2015) Slime mould foraging behaviour as optically-coupled logical operations, *International Journal of General Systems* DOI: 10.1080/03081079.2014.997528.

Martínez, G. J., Adamatzky, A., and Costello, B. L. (2008) On logical gates in precipitating medium: cellular automaton model, *Physics Letters A*, 1(48):1–5.

Martínez, G. J., Adamatzky, A., Morita, K., and Margenstern, M. (2010) Computation with competing patterns in Life-like automaton, In: *Game of Life Automata*, A. Adamatzky (ed.), Springer, chapter 27, pages 547–572.

Martínez, G. J., Adamatzky, A., and McIntosh, H. (2012) Computing on rings In: *A computable Universe: Understanding and Exploring Nature as Computation*, H. Zenil (ed.), World Scientific Press, chapter 14, pages 283–302.

Margolus, N. H. (1984) Physics-like models of computation, *Physica D*, 10(1-2):81–95.

Margolus, N. H. (1998) Crystalline Computation, In: *Feynman and computation: exploring the limits of computers*, A. J. G. Hey (Ed.), Perseus Books, 267–305.

Margolus, N. (2002) Universal Cellular Automata Based on the Collisions of Soft Spheres, In: *Collision-Based Computing*, A. Adamatzky (ed.), Springer, chapter 5, pages 231–260.

Martínez, G. J., Adamatzky, A., Stephens, C. R., and Hoeflich, A. F. (2011) Cellular automaton supercolliders, *International Journal of Modern Physics C*, 22(4):419–439.

McIntosh, H. V. (2009) *One Dimensional Cellular Automata*, Luniver Press, UK.

Mills, J. W. (2008) The Nature of the Extended Analog Computer, *Physica D*, 237(9):1235–1256.

Minsky, M. (1967) *Computation: Finite and Infinite Machines*, Prentice Hall.

Mitchell, M. (2001) Life and evolution in computers, *History and Philosophy of the Life Sciences*, 23:361–383.

Martínez, G. J., Morita, K., Adamatzky, A., and Margenstern, M. (2010) Majority adder implementation by competing patterns in Life-like rule B2/S2345, *Lecture Notes in Computer Science* 6079:93–104.

Martínez, G. J., McIntosh, H. V., Seck-Tuoh-Mora, J. C., and Vergara, S. V. C. (2011) Reproducing the cyclic tag system developed by Matthew Cook with Rule 110 using the phases $f_{1\_}1$," *Journal of Cellular Automata*, 6(2-3):121–161.

Martínez, G. J., Seck-Tuoh-Mora, J. C., and Zenil, H. (2013) Computation and Universality: Class IV versus Class III Cellular Automata, *Journal of Cellular Automata*, 7(5-6):393–430.

Margolus, N., Toffoli, T., and Vichniac, G. (1986) Cellular-Automata Supercomputers for Fluid Dynamics Modeling, *Physical Review Letters*, 56(16):1694–1696.

Schumann, A. and Adamatzky, A. (2011) Physarum Spatial Logic, *New Math. and Nat. Computation*, 7(3):483–498.

Gregory, L. S., Orlov, A. O., Amlani, I., Bernstein, G. H., Lent, C. S., Merz, J. L., and Porod, W. (1999) Quantum-Dot Cellular Automata: Line and Majority Logic Gate, *Japanese Journal of Applied Physics*, 38:7227–7229.

Tsuda, S., Aono, M., and Gunji, Y-P (2004) Robust and emergent Physarum logical-computing, *BioSystems*, 73:45–55.

Toffoli, T. (1998) Non-Conventional Computers, *Encyclopedia of Electrical and Electronics Engineering* (John Webster Ed.), 14:455–471, Wiley & Sons.

Toffoli, T. (2002) Symbol Super Colliders, In: *Collision-Based Computing*, A. Adamatzky (Ed.), Springer, chapter 1, 1–22.

Tuszyński, J. A., Portet, S., Dixon, J. M., Luxford, C., and Cantiello, H.F. (2004) Ionic wave propagation along actin filaments, *Biophysical journal*, 86:1890–1903.

Wolfram, S. (1984) Universality and complexity in cellular automata, *Physica D*, 10:1–35.

Wolfram, S. (2002) *A New Kind of Science*, Wolfram Media, Inc., Champaign, Illinois.

Wuensche, A. (2011) *Exploring Discrete Dynamics* Luniver Press, UK.

Slime mould actin sensoriactuation networks: topology, dynamic transformations and models of self-assembly

Richard Mayne and Andrew Adamatkzy

Unconventional Computing Group, University of the West of England, Bristol, BS16 7AS, UK
Richard.Mayne@uwe.ac.uk

Abstract

The plasmodium of slime mould *Physarum polycephalum* is a macroscopic multinucleate single cell which, despite displaying apparently 'intelligent' behaviour patterns, lacks any of the physical components usually associated with biological intelligence. In previous works, we have suggested that the plasmodial actin network functions as a nano-scale information processor whose functions contribute to the organism's ability to compute the solutions to complex functions. In this investigation, we produce experimental observations of plasmodial actin networks and use proximity diagrams to analyse their topology. Consequently, using optimised Toussaint Hierarchy and β-skeleton models, we approximate the characteristical dynamic topological transformations resulting from the spontaneous actin assembly/disassembly in areas with denser networks such as growth cones. We conclude by discussing the role of the cytoskeleton in facilitating intracellular computing with various media (vesicles, electrical potential, Davydov solitons), how computation can be implemented in such a network and practical considerations for designing 'useful' actin computing circuits. Our results emphasise the viability of biological substrates for unconventional computing and the value of proximity graphs in approximating features of living systems.

Keywords: *Physarum polycephalum*, slime mould, proximity graph, actin, unconventional computing.

Introduction

The cytoskeleton is a ubiquitous organelle in eukaryotic cells that comprises a 'scaffold' of proteins whose impermanent topology is highly dynamical. Where it was once thought that the primary function of the cytoskeleton was to provide mechanical support to the cell, it is now clear that it also participates in a multitude of intracellular processes, some of which are not yet fully characterised, including trafficking of organelles through the cytoplasm, facilitating cell movement and transduction of energetic/signalling events through the cell (Janmey, 1998). Although there are a great many varieties of protein that may be present in a cell's cytoskeleton, the predominant two are tubulin, which forms long tubular structures (microtubules), and actin, which forms smaller filamentous double helices (microfilaments).

Of the energetic events that may pass through the cytoskeleton, the best characterised is likely the transmission of mechanical force through the cell via actin microfilaments linked to cell-surface mechanoreceptors (Janmey, 1998). There is a growing body of evidence to suggest, however, that further forms of intracellular signal are carried by the cytoskeleton such as electrical potential, vesicle-bound signalling molecules and quantum events such as Davydov solitons (Carpenter, 2000; Craddock et al., 2012; Davydov, 1977; Forgacs et al., 2004; Maniotis et al., 1997; Schmidt and Hall, 1998; Tuszyński et al., 2004). It has been suggested by several authors that many of the emergent properties displayed by organisms (e.g. synergistic cooperation between cells in complex organ systems such as the brain) arise at the cellular level from myriad signalling processes travelling through and interacting with the cytoskeleton (we refer the reader to Mayne et al. (2014) for a review of the concept). If this is indeed the case, then by extension any research furthering our knowledge of cytoskeletal processes will enhance our understanding of poorly-characterised phenomena such as brain function and hence will precipitate a new wave of technologies and medical therapies (Hameroff, 1987; Priel et al., 2010).

From the perspective of Computer Science, the advancement of this topic is enthusing as biological signalling processes can, when interpreted in the language of computation, represent intracellular 'data' that are the consequences of environmental 'input' (from cell-surface receptors) which are transdcuced into a regular, repeatable format — arguably, by the cytoskeleton — that the cell can interpret and act upon. We have argued in previous publications that the cell utilises it's cytoskeleton as an intracellular network wherein data can be represented as any energetic event whose presence on a specific locus of a cytoskeletal protein is a logical '1' and *vice versa*: interactions between signals equate to computation whose output is a change in cellular behaviour caused by the interaction of a signalling event with a target effector (Mayne et al., 2014; Mayne and Adamatzky, 2014). Under this premise, we have proposed that the cytoskeleton is a highly desirable unconventional computing substrate in which reaction-diffusion or/and collision-based computing may be implemented.

Whilst we are not the first to comment on the putatively computational nature of cytoskeletal dynamics, the majority of the work to date on cytoskeletal computing focuses on microtubules and typically concerns modelling the cytoskeleton as a conventional general purpose computer. It has been previously demonstrated, for example, that microtubules may transmit propagating waves of transitions in protein conformational state, and in this way they may be thought of as a data bus; Boolean logical operations have been suggested to occur when these events interact with the microtubule-associated proteins that link microtubules to other cytoskeletal components (Craddock et al., 2012; Lahoz-Beltra et al., 1993). We emphasise, conversely, that the functionality of a cell is, whilst analogous to a conventional computer in some aspects, so divergent from silicon-based architectures that any direct comparisons between the two are null; consequently, practical computing devices based on biological substrates must be built on emphatically unconventional paradigms.

In previous works, we have formalised actin network topology in a single-celled organism, slime mould *Physarum polycephalum*, with a range of proximity graphs with different network connectivities and suggested that it may switch between graph type based on momentary physiological needs (Mayne et al., 2014; Mayne and Adamatzky, 2014); as such, it may dynamically alter the properties of its own intracellular computation network to suit its needs.

Slime mould *P. polycephalum* is a plasmodial ('true') slime mould that resembles a giant amoeba whilst in its vegetative life cycle phase (Fig. 1) (Stephenson and Stempen, 1994). Possessing many millions of nuclei and the intracellular machinery to migrate at a comparatively rapid rate, plasmodial slime moulds are essentially giant eukaryotic cells and are hence valuable model organisms. Intense interest has been generated in slime mould as an unconventional computing substrate as certain behaviour patterns it exhibits — labyrinth navigation (Nakagaki, 2001), use of an extracellular chemical memory (Reid et al., 2012), optimisation of nutrient harvesting networks (Adamatzky, 2010) etc. — may reasonably be described as apparently 'intelligent'. But how is this possible in an organism with no brain or neural tissue? The aforementioned 'cytoskeletal-basis of emergent behaviour' hypothesis is an attractive explanation to this research question, further indicating the suitability of slime mould for such a study.

The plasmodial cytoskeleton is not as well characterised as those of mammalian cells, but the constituent proteins are highly conserved: slime mould microfilaments, as with their mammalian counterparts, are composed of globular actin monomers (g-actin) whose quarternary structure is a double helix, individual fibres of which are known as f-actin (Fig. 2). In mammalian cells, the actin network spans the entire cell but is most concentrated in a dense cortical region about the cell's periphery, where it articulates onto membrane-bound proteins. This is also true for slime mould, although we have previously found that the organism's advancing anterior margin, which is formed from the confluence of a multitude of pseudopodia, contains copious actin in a highly dense, interconnected network (Mayne et al., 2014). Whilst pseudopodium formation via momentary assembly of actin is a well-observed phenomenon, this posits the question of how this topological dimorphism impacts on intracellular computation.

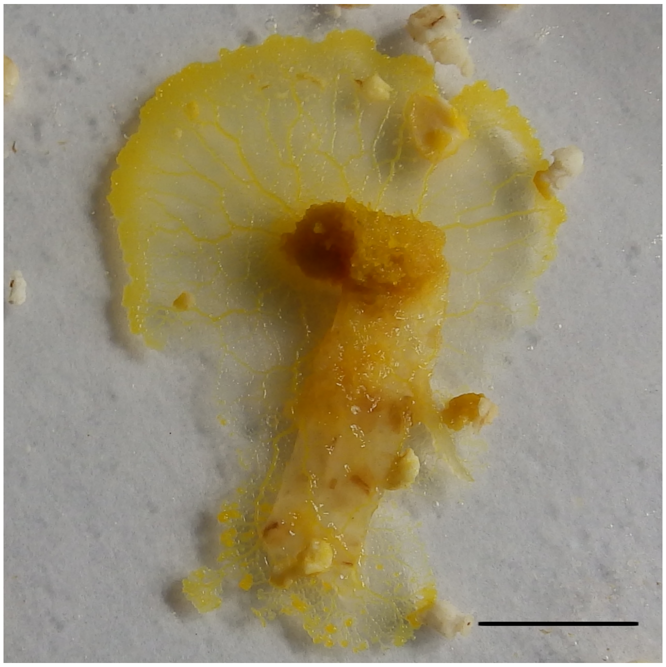


Figure 1: Photograph of *P. polycephalum* plasmodium growing on an agar plate consuming oat flakes. Note how the anterior margin is 'fan-shaped', but the rest of the organism is predominantly formed from tubular structures. Scale bar 10mm.

In this investigation, we present our revised methods for extracting and formalising dynamic, multi-topology actin network topologies and expand with models for cytoskeletal growth and experimental data demonstrating how practical mixed-topology cytoskeletal computing circuits may be implemented. We conclude by discussing practical aspects of cytoskeletal circuit design.

Methods

Stock plasmodia of *P. polycephalum* (strain HU554 × HU560) were cultivated on 2% non-nutrient agar (NNA) plates at room temperature in the absence of light. They were provided with porridge oats as a nutrient substrate and were sub-cultured routinely every 3–4 days, as required. Samples were prepared for fluroescent-labelling of actin as

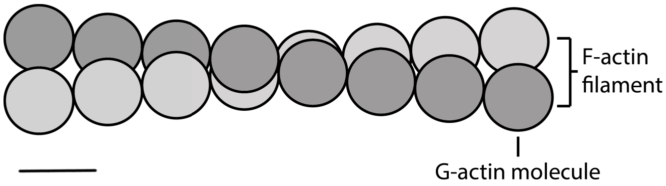


Figure 2: Schematic diagram of an actin microfilament as a double helix structure composed of individual G-actin monomers. Scale bar approx. 5nm, helix twist not to scale. Adapted with permission from Adamatzky and Mayne (2015).

follows:

Two 'islands' of NNA were prepared on large glass microscope coverslips by dripping approximately 2×0.5ml of molten agar with a pipette, with a gap of approximately 10mm separating the two. Samples of stock plasmodium were homogenised with a scalpel blade and transferred to one NNA island. An oat flake was placed on the second island and the the cover slip was placed in an air-tight Petri dish, which was left in the dark at room temperature to propagate. Samples were routinely checked and chemically fixed when the plasmodium was observed propagating across the gap towards the second island; samples of the anterior margin were taken by fixing before the organism had reached the second island, whereas tubules were prepared by waiting until it had colonised the second island.

Fixation was achieved by flooding the Petri dish with 2% paraformaldehyde in pH 7.2 phosphate-buffered saline solution for 1 hour. This was followed by 3×5 minute rinses in the same buffer, after which the sample was permeabalised with 0.1% Triton X-100. Following further rinsing and draining, Alexa Fluor 488 Phalloidin (Molecular Probes, USA) was added for 1 hour at a concentration of 5μl in methanol. The samples were then washed again before being stained with DAPI (Abcam, UK).

Confocal microscopy was performed with a Perkin Elmer UltraView ERS FRET-H spinning disk confocal laser scanning microscope. For details of image post-processing, please see the Appendices.

Results

Visualisation of actin

Exemplar confocal micrographs of the plasmodial actin network are shown in Figs. 3(A,B) and 4(A,B), which represent a fragment of plasmodial tube and psuedopodia from the anterior margin, respectively. In corroboration with our previous findings, the plasmodial actin network was observed to be concentrated more in the cell's cortical regions and pseudopodia.

Formalisation of plasmodial actin networks

Plasmodial actin network topology may be derived from proximity graphs if edges are represented by microfilaments and nodes by nuclei. Nuclei were chosen as vertices due to their representing anchor points at which microfilaments terminate. Furthermore, whilst the functionality of the plasmodium's nuclei cannot be compared with that of any conventional computing component, they are likely to be recipients and initiators of some forms of cytoskeletal signalling (receptor-nucleus, nucleus-nucleus or/and nucleus-effector). Proximity graphs were generated from confocal micrographs by manually extracting vertices (Fig. 3(B–E), 4(B,C).

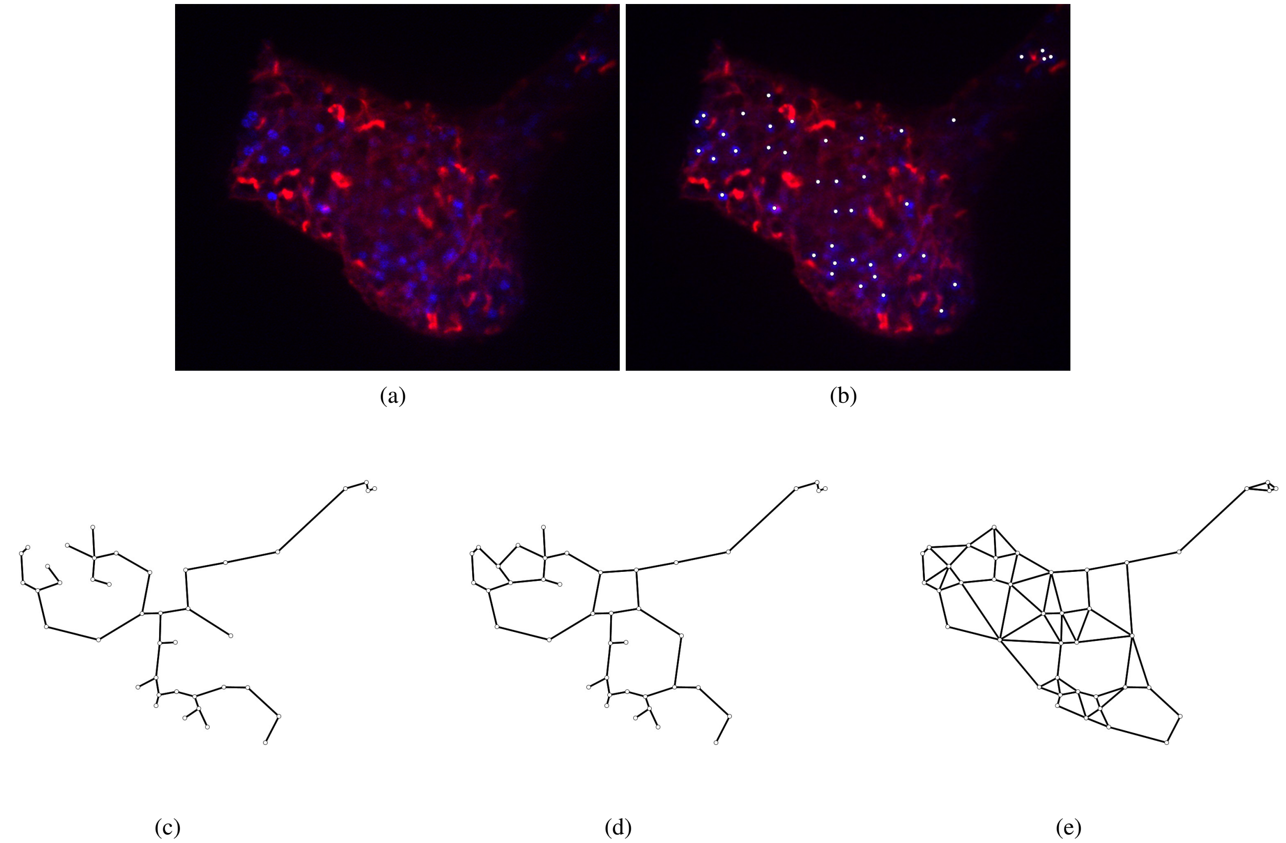
A planar graph consists of nodes which are points of the Euclidean plane and edges which are straight segments connecting the points. A planar proximity graph is a planar graph where two points are connected by an edge if they are proximate (spatially close) in some sense. A pair of points is assigned a certain neighbourhood, and points of the pair are connected by an edge if their neighbourhood is empty. Here we consider the most common proximity graph as follows.

- $\mathbf{GG}$: Points a and b are connected by an edge in the Gabriel Graph $\mathbf{GG}$ if disc with diameter $dist(a,b)$ centred in middle of the segment ab is empty (Gabriel and Sokal, 1969; Matula and Sokal, 1984).
- $\mathbf{RNG}$: Points a and b are connected by an edge in the Relative Neighbourhood Graph $\mathbf{RNG}$ if no other point c is closer to a and b than $dist(a,b)$ (Toussaint, 1980).
- $\mathbf{MST}$: The Euclidean minimum spanning tree (MST) is a connected acyclic graph which has minimum possible sum of edges' lengths.

In general, the graphs relate as $\mathbf{MST} \subseteq \mathbf{RNG} \subseteq \mathbf{GG}$ (Jaromczyk and Toussaint, 1992; Matula and Sokal, 1984; Toussaint, 1980); this is called the Toussaint hierarchy.

Thus, we can speculate that depending on momentary actual demands of the slime mould's physiology the topology of its communication network can rapidly change between a proximity graph with a high number of connections between nodes (Gabriel graph), the cyclic graph with minimal number of links (relative neighbourhood graph), and the acyclic proximity graph (spanning tree). This can be demonstrated in Figs. 3 and 4, wherein a posterior fragment of plasmodial tube was observed to be better represented by the less-connected proximity graphs than advancing pseudopodia, which favour a more interconnected topology.

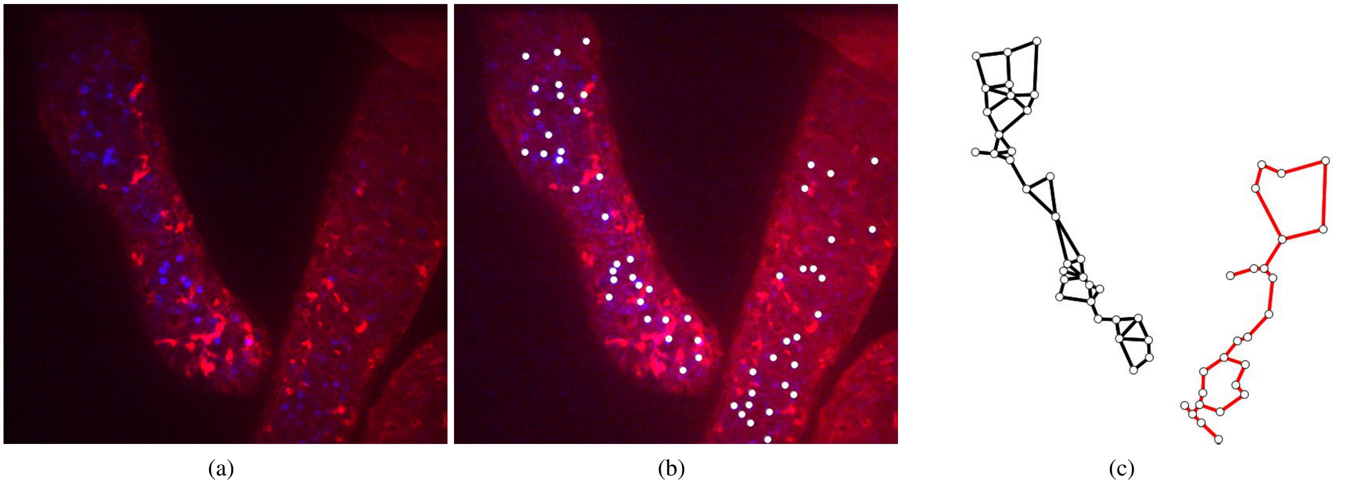
The topological transformations between the proximity graphs may also be expected during morphological transformations of the slime mould's active growing zones. This can be demonstrated in Fig. 4, where two adjacent pseudopodia demonstrate differing topologies, despite being connected to



(a) (b)

(c) (d) (e)

Figure 3: Actin network and corresponding proximity graphs from a 30μm optical section from an intact plasmodial vein fragment situated c. 5mm from the anterior margin, where nuclei are represented by vertices. Note how the actin network is best approximated by the more minimalistic proximity graphs. Original magnification x400. (a) Confocal micrograph, red = actin, blue = nuclei. (b) Graph vertices plotted onto nuclei. (c) Spanning tree. (d) Relative neighbourhood graph. (e) Gabriel graph.



(a) (b) (c)

Figure 4: Actin networks within two adjacent pseudopodia and corresponding proximity graphs in a 30μm optical section, where nuclei are represented by vertices. The network is visibly denser in the left-hand branch as it terminates in a growth cone: as such, each should arguably be represented by different varieties of graph, despite their both being part of the same anatomical region. Accordingly, each are represented by different graph types. Original magnification x400. (a) Confocal micrograph, red = actin, blue = nuclei. (b) Graph vertices. (c) Black network = Gabriel graph, red network = relative neighbourhood graph.

the same network region. As such, they are best approximated by different varieties of proximity graph.

Discussion

Sensoriactuation networks for intracellular computing

If it is assumed that each actin strand is capable of transmitting information, a cell's computational 'ability' is, by extension, proportional to the abundance and interconnectedness of it's actin network, where 'ability' here is abstracted to encompass the informational capacity of the network, speed/efficiency of communication, number of logical functions per unit area, etc. This implies, therefore, that more data transduction, transmission and interactions — more *computing* — occurs in active growth zones of the slime mould actin sensoriactuation network, i.e. where network density is highest. Conversely, in less-active anatomical locations, actin networks are more diminutive (presumably to conserve energy), where less data transduction/motive power are required.

Whilst this is a somewhat obvious point to highlight (at least from an evolutionary perspective), let us consider how intracellular computation may be influenced by data network topology. Computing within the more interconnected graph varieties, especially Gabriel graph-based methods, has been demonstrated to be viable for instance-based learning algorithms such as majority voting (Toussaint, 1980; Toussaint and Berzan, 2012). We may speculate, therefore, that analogous 'biological algorithms' are integrated via actin data streams whose observable output are synchronised, effectively autonomous behaviours. An example is determination of tip growth direction: the cell's cortical actin becomes innervated via multiple chemoreceptors which causes many signals to be transduced into the local network. By the majority vote, patterns of interference are negated, allowing the organism to better synchronise its response into a coherent action (directional tip growth). By comparison, actin filaments in the less-interconnected regions — which tend to contain longer filaments known as stress fibres (Mofrad and Kamm, 2006) — may act in a manner more akin to data buses.

This is an interesting perspective as it implies that the parallelism inherent in biological computing substrates is automatic and occurs without the need for expending energy on synchronisation. As such, the complexity of the organism is reduced whilst enabling the spontaneous generation of complex behaviours — in effect, this is a definition of emergence. These observations are complimentary to recent advances in the field of morphological computation and entity embodiment which state that 'outsourcing' a certain amount of computational work to the morphology of data streams is essential in the design of artificially intelligent entities (Hauser et al., 2012; Lungarella and Sporns, 2006). This concept is inimically linked to data 'structuring' — transducing and transmitting signals in a repeatable and unambiguous manner — which is another important concept in morphological computation and control theory (Füchslin et al., 2013; Hauser et al., 2012). By logical extension, this would seem to suggest that the cytoskeleton is a medium for structuring sensorimotor data streams and hence that the emergent behaviours displayed by cells may be a product of cytoskeletal processes.

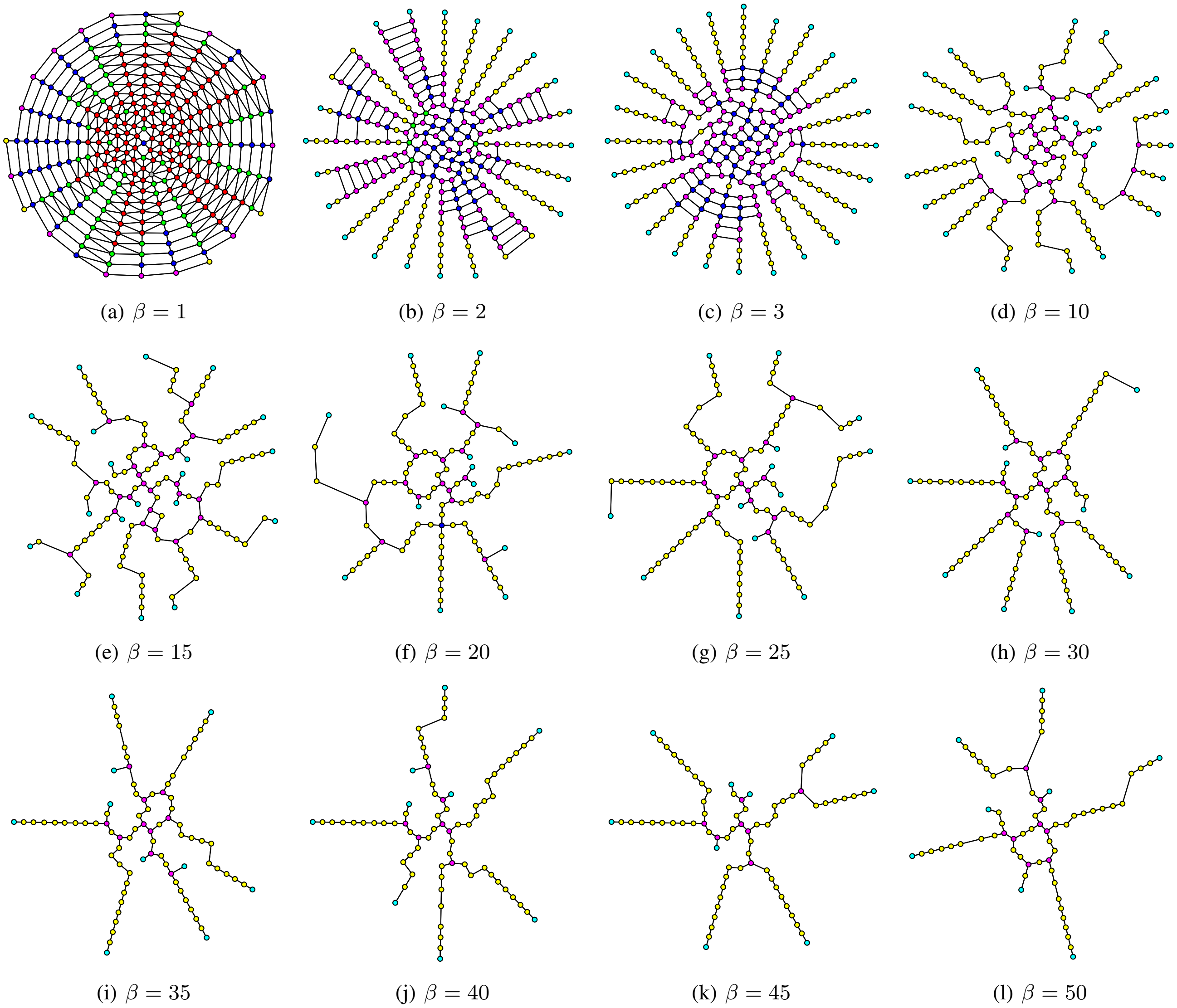
Modelling network growth

We propose that actin network dynamic transformations may be approximated by β-skeletons (Fig. 5). Given a set $\mathbf{V}$ of planar points, for any two points p and q we define a β-neighbourhood $U_\beta(p,q)$ as the intersection of two discs with radius $\beta|p-q|/2$ centered at points $((1-\frac{\beta}{2})p, \frac{\beta}{2}q)$ and $(\frac{\beta}{2}p, (1-\frac{\beta}{2})q)$, $\beta \geq 1$ (Jaromczyk and Toussaint, 1992; Kirkpatrick and Radke, 1985). Nodes p and q are connected by an edge in β-skeleton if the pair's β-neighbourhood contains no other nodes from $\mathbf{V}$. In the hypothetical scenario shown in Fig. 5 we imitate transformation of a centrifugal network by tuning the neighbourhood parameter β from 1 to 50; see details of the algorithm to grow β-skeletons in Adamatzky (2013). The transformation is implemented via pruning of redundant links and gradual formation of the acyclic graphs. The pruning starts at the peripheral parts of the graph, see e.g. Fig. 5(D,E), and propagates towards the central core of the graph, see e.g. Fig. 5(F,G), until the whole graph is transformed into an —almost — acyclic graph, as shown in Fig. 5(L).

Practical actin computing

Let us consider the biophysical events that are transmitted down actin fibres. Actin, along with its companion motor protein myosin, is instrumental in the retention and transport of vesicle-bound biomolecules and minerals (DePina and Langford, 1999); the actin network could therefore be a medium for implementing collision-based computing with vesicles. The same is also true for quantum events such as solitons and coherent waves of actin contraction. Electrical potential is, as previously alluded to, transmitted down actin fibres in the form of ionic waves and as such, the relative concentrations of relevant ions could be used for implementing a variety of computing paradigms.

To engineer practical actin computing circuits we are presented with a few technical considerations. First is to generate an actin network in a desired topology; this may be *in vitro* or *in vivo*. Although minute manipulation of actin fibres has been demonstrated *in vitro* (Lin and Cantiello, 1993), we suggest that actin growth within model organisms such as *P. polycephalum* is also a viable method to this end as growth patterns are essentially programmable by manipulation of the organism's environment and internal feedback mechanisms. The topology of such networks can be predicted with the models presented here and dynamic trans-



(a) $\beta = 1$ (b) $\beta = 2$ (c) $\beta = 3$ (d) $\beta = 10$

(e) $\beta = 15$ (f) $\beta = 20$ (g) $\beta = 25$ (h) $\beta = 30$

(i) $\beta = 35$ (j) $\beta = 40$ (k) $\beta = 45$ (l) $\beta = 50$

Figure 5: Modelling experiments approximating topological transformations of actin sensoriactuation networks in *P. polycephalum* represented by β-skeletons. Evolution is determined by parameter β and node degrees are indicated by their colours. By decreasing the β parameter, the network can be observed to erode from a highly interconnected network to a minimalistic topology. See Adamatzky (2013) for further details.

formations between different topologies (through network dilation/erosion) should be used to achieve the desired connectivity — ideally through natural mechanisms (nucleation factors, inhibitors, etc.).

The second consideration is the initiation and synchronisation of informational events. Again, this issue may best be addressed by the manipulation of environmental and internal factors, but may also be induced artificially. In slime mould models, for example, actin network contraction can be initiated by tactile stimulation. The interaction environment's topology plays an important role in synchronisation, especially in collision-based models as factors such as filament length and delay elements are necessarily functions of morphology.

The findings presented here indicate that actin is an extremely valuable unconventional computing substrate.

Conclusion

A prime motivation into researching biological ccomputing substrates such as slime mould is their amorphism — a property that implies architecture-less massive parallel processing. *P. polycephalum's* ostensibly emergent computing capabilities may be a product of its cytoskeletal topology and the information processing events therein. This implies that emergent behaviour is a product the physical properties of the data network; this has important implications for our understanding of concepts such as human intelligence, which is commonly regarded as a product (in part, at least) of neural network morphology and the characteristics of the juctions (synapses) between them. In future works, we will expand this concept with experimental implementations of actin-based computing.

Conflict of Interest

The authors declare no conflict of interest and gratefully acknowledge funding from the European Commission Seventh Framework Program.

Image Processing

Photographs were taken with a Samsung I9300 digital camera. Confocal micrographs were digitally post-processed in Volocity (Improvision, UK), and received colour assignment, deconvolution and brightness/colour adjustments. Unprocessed image files will be made available on request.

Acknowledgements

The authors extend their thanks to Ben De Lacy Costello for his supervision and technical support with the experimental work undertaken.

References

Adamatzky, A. (2010). *Physarum machines: Computers from slime mould*. World Scientific Publishing, London.

Adamatzky, A. (2013). On growing connected beta-skeletons. *Computational Geometry*, 46(6):805–816.

Adamatzky, A. and Mayne, R. (2015). Actin Automata: Phenomenology and Localizations. *International Journal of Bifurcation and Chaos*, 25(2).

Carpenter, C. (2000). Actin cytoskeleton and cell signalling. *Critical Care Medicine*, 28(4):94–99.

Craddock, T., Tuszyński, J., and Hameroff, S. (2012). Cytoskeletal signalling: is memory encoded in microtubule lattices by camkii phosphorylation? *PLoS Computational Biology*, 8(3).

Davydov, A. (1977). Solitons and energy transfer along protein molecules. *The Journal of Theoretical Biology*, 66:379–387.

DePina, A. and Langford, G. (1999). Vesicle transport: the role of actin filaments and myosin motors. *Microscopy Research and Techniques*, 47(2):93–106.

Forgacs, G., Yook, S. H., Janmey, P., Jeong, H., and Burd, C. (2004). Role of the cytoskeleton in signalling networks. *The Journal of Cell Science*, 117(3):2769–2775.

Füchslin, R., Dzyakanchuk, A., Flumini, D., Hauser, H., Hunt, K., Luchsinger, R., Reller, B., Scheidegger, S., and Walker, R. (2013). Morphological computation and morphological control: steps towards a formal theory and applications. *Artificial Life*, 19:9–34.

Gabriel, K. and Sokal, R. (1969). A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278.

Hameroff, S. (1987). *Ultimate Computing: biomolecular consciousness and nanotechnology*. Elsevier, Amsterdam, first edition.

Hauser, H., Ijspeert, Füchslin, R., Pfeifer, R., and Maass, W. (2012). Towards a theoretic foundation for morphological computation with compliant bodies. *Biological Cybernetics*, 105:355–370.

Janmey, P. (1998). The cytoskeleton and cell signalling: component localization and mechanical coupling. *Physiological Reviews*, 78(3):763–781.

Jaromczyk, J. and Toussaint, G. (1992). Relative neighbourhood graphs and their relatives. In *Proceedings of the IEEE*, volume 80, pages 1502–1517.

Kirkpatrick, D. and Radke, J. (1985). *A framework for computational morphology*. IBM, Amsterdam.

Lahoz-Beltra, R., Hameroff, S., and Dayhoff, J. (1993). Cytoskeletal logic: a model for molecular computation via boolean operations in microtubules and microtubule-associated proteins. *Biosystems*, 29(1):1–23.

Lin, E. C. and Cantiello, H. F. (1993). A Novel Method to Study the Electrodynamic Behavior of Actin Filaments. Evidence for Cable-like Properties of Actin. *Biophysical Journal*, 65(December 1992).

Lungarella, M. and Sporns, O. (2006). Mapping information flow in sensorimotor networks. *PLoS Computational Biology*, 2(10).

Maniotis, A., Chen, C., and Ingber, D. (1997). Demonstration of mechanical connections between integrins, cytoskeletal filaments and nucleoplasm that stabilise nuclear structure. *PNAS*, 94(3):849–854.

Matula, D. and Sokal, R. (1984). Properties of gabriel graphs relevant to geographical variation research and clustering of points in the same plane. *Geographical Analysis*, 12:205–222.

Mayne, R. and Adamatzky, A. (2014). The Physarum polycephalum actin network : formalisation , topology and morphological correlates with computational ability. In *8th International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pages 87–94.

Mayne, R., Adamatzky, A., and Jones, J. (2014). On the role of the plasmodial cytoskeleton in facilitating intelligent behaviour in slime mould *Physarum polycephalum*. *Communicative and Integrative Biology*, 7(1):e32097.

Mofrad, M. and Kamm, R. (2006). *Cytoskeletal mechanics: models and measurements in cell mechanics*. Cambridge University Press, Cambdridge, first edition.

Nakagaki, T. (2001). Smart behavior of true slime mold in a labyrinth. *Research in Microbiology*, 152(9):767–770.

Priel, A., Tuszynski, J., and Woolf, N. (2010). Neural cytoskeleton capabilities for learning and memory. *Journal of Biological Physics*, 36(1):3–21.

Reid, C. R., Latty, T., Dussutour, A., and Beekman, M. (2012). Slime mold uses an externalized spatial ”memory” to navigate in complex environments. *Proceedings of the National Academy of Sciences of the United States of America*, 109(43):17490–4.

Schmidt, A. and Hall, M. (1998). Signalling to the actin cytoskeleton. *Annual Reviews of Cell and Developmental Biology*, 14:305–338.

Stephenson, S. and Stempen, H. (1994). *Myxomycetes: a handbook of slime molds*. Timber Press, Oregon.

Toussaint, G. (1980). The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12:261–268.

Toussaint, G. and Berzan, C. (2012). Proximity-graph instance-based learning, support vector machines, and high dimensionality: An empirical comparison. In Perner, P., editor, *Machine Learning and Data Mining in Pattern Recognition*, volume 7376 of *Lecture Notes in Computer Science*, pages 222–236. Springer Berlin Heidelberg.

Tuszyński, J., Portet, S., Dixon, J., Luxford, C., and Cantiello, H. (2004). Ionic wave propagation along actin filaments. *Biophysical Journal*, 86:1890–1903.

Mimicking the Exploration of 3D Terrains by Physarum with Cellular Automata Models

Michail-Antisthenis I. Tsompanas and Georgios Ch. Sirakoulis

Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi 67100, Greece
mtsompan@ee.duth.gr, gsirak@ee.duth.gr

Abstract

The plasmodium of *Physarum polycephalum* has been used to solve graphically represented puzzles and mimic the development of different types of networks in laboratory experiments. Thus, it can be characterized as a biological computing device. Although this non-silicon computer is mainly using input data solely coded as the distances between points of interest, some other environmental factors seem to have an effect on the behaviour of the plasmodium and, thus, the produced results. These factors can be elevations or illumination of the experimental surface. This paper presents a model based on Cellular Automata (CA) that imitates the results provided by the living substrate which is subjected to the effects of the elevations of the 3-dimensional (3D) experimental surface. Developing software based models is based on their robustness compared to the time-expensive real-life biological computing device. Some trivial experiments are presented to depict the agreement between the resultant networks developed by the simulated and the real plasmodium.

Introduction

The vegetative stage of acellular slime mould *Physarum polycephalum*, namely plasmodium, is a multi-nuclei single cell, which is, recently, fairly entitled as a biological computer (Adamatzky, 2010). Due to the simplicity of its physiology and trivial cultivation and handling, it is broadly accepted and commonly used as a living substrate for biological/analog computing (Mayne et al., 2015). The plasmodium of *P. polycephalum* is used in carefully planned and conducted laboratory experiments that unveil its computing capabilities on geometrically represented problems (Adamatzky and Jones, 2015).

The plasmodium, which feeds on microscopic particles, interconnects nutrient sources (NSs) located into its vicinity with a tubular network. The resultant network has the functionality of transporting nutrients and chemical signals throughout the body of the plasmodium. As a great amount of the plasmodium's protoplasmic mass is concentrated over NSs to digest the available nutrients, the network needs to be of short total length and follow a minimum risk path. Consequently, the aforementioned ability of the plasmodium to draw an efficient network to survive in hostile environments, can be interpreted as computing capacity.

As a result, the biological computer, consisted of the plasmodium of slime mould, is introduced with input data that are represented as the topology of NSs. The results of the computation are manually identified with the interpretation of the network developed by the plasmodium, spanning all the NSs. However, the input data can be of higher complexity, as the plasmodium was identified as gravisensitive, positively geotropic (Block et al., 1986, 1998) and photo-phobic (Ueda et al., 1988; Nakagaki et al., 2007). Thus, by using elevations or illumination in parts of the laboratory experiment surface, the biological computer acts not only by taking into account the distances between points of interest (NSs) but some other important factors, too.

Given the fact that the laboratory experiments are rather time consuming and the plasmodium, as a living substrate, rarely reproduces the exact same results, software based models are proposed. The selection of Cellular Automata (CA) as the mathematical tool of the proposed model, is based on the fact that the plasmodium does not possess a central nervous system, albeit, has a distributed decision-making mechanism. Thus, the local rule describing the evolution of the states in CA is an ideal modeling candidate. Moreover, the movement of the plasmodium towards sources of attraction is considered to occur due to a massive distributed array of membrane-bound sensor proteins that covers the external boundaries of the plasmodium (Glockner et al., 2008). That distributed sensing mechanism is easily represented by CA and coherent with the emerging of global behavior from local interactions.

Furthermore, plasmodial sensing is continuous and processes of each receptor happen simultaneously. Thus, the inherent parallel nature of CA is a key characteristic for simulating the plasmodium. Nonetheless, that characteristic can further accelerate the proposed model by implementing it to parallel computers or dedicated hardware (Sirakoulis and Adamatzky, 2014). The proposed model is designed to imitate the foraging behaviour and network design of slime mould when first been starved and then introduced to a surface that has elevations and several distributed NSs. The proposed CA model was inspired by the CA model presented by Tsompanas and Sirakoulis (2012). An updated version was suggested, that delivered more realistic results compared with the real motorways and the results from the biological experiments (Tsompanas et al., 2014a,b). Moreover, the hardware implementation of the aforementioned models was also proposed (Tsompanas and Sirakoulis, 2012; Dourvas et al., 2015; Tsompanas et al., 2014b).

This paper is organized as follows. In the next section some related work is presented. After that the basic CA theoretical background is presented. Moreover, the details on the proposed model and basic results that reveal its applicability are provided. Finally, conclusions and some future work aspects are drawn in the last section.

Related Work

Several scientists who have conducted laboratory experiments with slime mould, have introduced some weight factors as input data in addition to the topology of NSs. These factors were represented by elevations of the terrain and inhomogeneous illumination of the surface. Some recent studies using such configurations are briefly described in the following.

Nakagaki et al. (2007) have presented a thorough study of the effect that light density has to the colonization policy of the plasmodium of *P. polycephalum.* In a simple rectangular experimental surface where the plasmodium has been left to spread, two NSs have been inserted at two opposite sites. The plasmodium was expected to connect them using a straight line. The control experiment has been conducted with a homogeneous light field. To study the effects that the photo-phobic characteristic of plasmodium has on the resultant network, more experiments have been conducted using inhomogeneous light fields with several illumination intensities. The results have been analyzed and the finding that the plasmodium tries to follow the minimum risk path was realized. Also, a mathematical model has been proposed to recreate the results.

Tero et al. (2010) have used the topology of cities in the area around Tokyo in a laboratory experiment to compare the results with the man-made rail network. They have observed that the results obtained by slime mould are not identical to the real network and realized that there are some geomorphology constrains in building a rail network such as mountains or lakes. As a result, they have decided to impose the same constrains to the plasmodium by using a variable light intensity to the experimental surface, in analogy with the mountains around Tokyo. That have resulted to networks with greater resemblance to the man-made rail network.

Adamatzky (2013) has studied the effects of network development by the plasmodium in terms of elevations on the surface of the laboratory experiment. The routes followed by immigrants from Mexico to the USA have been mimicked by the analogous modeler with the usage of 3D Nylon terrains of USA. Nonetheless, to highlight the significance of elevations to the foraging strategy and network development of the plasmodium, along with the 3D terrain, laboratory experiments with flat surfaces have been conducted as control. As a concluding remark, the author has suggested that the proposed configuration can imitate the human movement around elevations and can inspire algorithms for path planing robots.

Finally, Adamatzky (2014) has used 3D terrains equivalent to the morphology of USA and Germany as laboratory experiment surfaces, to evaluate man-made motorways, namely route 20, the longest road in USA, and autobahn 7, the longest national motorway in Europe. In this thorough study, the routes provided by laboratory experiments have been compared in terms of total length with the exact paths that the real man-made motorways follow. Moreover, laboratory experiments with and without ending points and with 3D terrains and flat surfaces have been conducted and their results have been presented and analyzed. Nonetheless, a computer simulation model have been proposed to imitate the road planing performed by slime mould and the paths provided by the model with different configurations have been presented.

Cellular Automata Basics

Cellular Automata (CA) are an idealization of a physical system in which space and time are discrete, and the physical quantities take only a finite set of values. CA were originally proposed by John von Neumann, who presented them as formal models of self reproducing organisms that can capture the essential features of systems where global behaviour arises from the collective effect of simple components which interact locally (von Neumann, 1966). Non-trivial CA are obtained whenever the dependence on the values at each site is non-linear. As a result, any physical system satisfying differential equations may be approximated by a CA, by introducing finite differences and discrete variables (Sirakoulis et al., 2000). A CA consists of a regular grid of cells. Each cell can take, not simultaneously, k different states, where k is a finite number, equal or greater than 2. Cells update their states in discrete time. That means that the state of each cell in the lattice changes only at discrete moments of time, namely at time steps t. The time step $t = 0$ is usually considered as the initial step and therefore no changes at the state of the cells occur.

For each cell, a set of cells called its neighbourhood (usually including the cell itself) is defined relative to the specified cell (Adamatzky, 1994). Regarding the two-dimensional (2D) CA, there are two fundamental types of neighbourhoods that are mainly considered:

- *von Neumann* neighbourhood, that consists of the central cell, whose condition is to be updated, and the four cells located to the north, south, east and west of the central cell

- *Moore* neighbourhood, that consists of the same cells with the *von Neumann* neighbourhood together with the four other adjacent cells of the central cell (i.e. north-west, north-east, south-east and south-west cells).

The evolution of the cells demands the definition of the state of each cell, as well as of the local transition function:

The state of a CA is described by a set

$$S(\vec{r},t) = \{S_1(\vec{r},t), S_2(\vec{r},t), ..., S_m(\vec{r},t)\} \quad (1)$$

of variables, that connects with each position $\vec{r}$ of the array and expresses the local internal state of each cell at time step $t = 0, 1, 2, ...$ The local transition function is defined as:

$$R = \{R_1, R_2, ..., R_m\} \quad (2)$$

and determines the evolution during time, of the internal state of each cell according to the following equation:

$$S_p(\vec{r},t+1) = R_p\left(S(\vec{r},t), S(\vec{r}+\vec{\delta}_1,t), ..., S(\vec{r}+\vec{\delta}_m,t)\right) \quad (3)$$

where the position $\vec{r}+\vec{\delta}_k,\ k \in \{1,...,m\}$ describes the neighbouring cells of each $\vec{r}$ cell.

CA have sufficient expressive dynamics to represent complex phenomena and, at the same time, can be simulated exactly by digital computers because of their intrinsic discreteness, i.e. the topology of the simulated object is reproduced in the simulating device (Vichniac, 1984; Mardiris et al., 2008). Prior and more recent works proved that CA are very effective in simulating physical systems and solving scientific problems, because they can capture the essential features of systems where global behaviour arises from the collective effect of simple components, which interact locally (Feynman, 1982; Wolfram, 1986; Sirakoulis and Bandini, 2012). Furthermore, they can easily handle complicated boundary and initial conditions, inhomogeneities and anisotropies (Sirakoulis et al., 2000).

The CA approach is consistent with the modern notion of unified space-time. In computer science, space corresponds to memory and time to processing unit. In CA, memory (CA cell state) and processing unit (CA local rule) are inseparably related to a CA cell (Sirakoulis et al., 2003; Progias and Sirakoulis, 2013). Finally, CA can be easily coupled with other computational tools so as to significantly enhance their performance and extend their applications field (Werfel et al., 2000; Ashlock and McNicholas, 2013).

Models based on CA lead to algorithms which are fast when implemented on serial computers because they exploit the inherent parallelism of the CA structure. These algorithms are also appropriate for implementation on massively parallel computers (Spezzano et al., 1996), such as the Cellular Automaton Machine (CAM) (Wilding et al., 1991) or Field Programmable Gate Arrays (FPGAs) (Mardiris et al., 2008; Georgoudas et al., 2010; Jendrsczok et al., 2009).

Proposed Model

The proposed model is designed to imitate the foraging behaviour and connecting strategy of slime mould in laboratory experiments. During these experiments, slime mould is inoculated on a NS and then introduced to an area where several NSs are placed in key positions. The novelty of this model resides in the fact that the possible elevations of the experimental surface have been taken into account. As several studies proposed that there is a significant change in the resultant network when there are elevations on the terrain, previously suggested models (Tsompanas et al., 2014a,b) are updated.

Although the actual laboratory experiments are conducted on 3D terrains (Adamatzky, 2014) the CA grid used here is two dimensional. The elevations of the terrain are represented with a constant value on the state ($S^t_{(i,j)}$) of each cell $C_{(i,j)}$. More specifically the parameters constituting the state of each cell are the following:

- RS is the parameter illustrating if the corresponding cell is within the **reachable surface** by the plasmodium,
- ALT is the parameter depicting the **altitude** of the surface represented by every cell,
- AC is the parameter corresponding to the **attractants concentration** in every cell emitted by the NSs,
- MC is the parameter corresponding to the protoplasmic **mass concentration** in every cell and
- TU is the parameter illustrating if the corresponding cell is a part of the **tubular** network of the slime mould.

Note here that ALT parameter is taking values in absolute units, to represent a relative difference of the elevations of adjacent cells. Consequently, the state of every cell $C_{(i,j)}$ at a given time step t is given in Eq. 4. The neighbourhood type used for the proposed model is Moore neighbourhood, which is described in the previous section.

$$S^t_{(i,j)} = \left[RS_{(i,j)}, ALT_{(i,j)}, AC^t_{(i,j)}, MC^t_{(i,j)}, TU^t_{(i,j)}\right] \quad (4)$$

The cells of the CA grid represent the whole surface of the laboratory experiment. Thus, the cells represent unreachable (cells in set U) as long as reachable (cells in set R) by the plasmodium surface. Moreover, each NS is represented by one cell (cells in set N). The point where the plasmodium is initially introduced to the experiment surface (starting point or SP) is, also, represented by one cell (cell in set I). Given all the aforementioned equation (5) can be defined.

$$N \subset R,\ I \subset R,\ R \cap U = \emptyset \quad (5)$$

The RS parameter has a corresponding value depending on the cell's set (as depicted in Eq. 6).

$$RS_{(i,j)} = \begin{cases} 1, & \forall i,j : C_{(i,j)} \in R \\ 0, & \forall i,j : C_{(i,j)} \in U \end{cases} \tag{6}$$

The model is analyzed in the following sequence of procedures:

1. Initialization of the model. The parameters of the diffusion equations are set and the topology of the SP and the NSs is introduced to the model.

2. Apply the diffusion equations for 50 time steps (t).

3. Check if any of the NSs is covered with a predefined percentage of MC (TMC). If there is at least one NS covered continue, else go to 2).

4. All NSs covered with TMC, are encapsulated by the plasmodium and therefore connected to a SP.

5. The NSs mentioned in 4) change into SPs, meaning their MC is set to 100. If no more than 5,000 time steps have passed ($t < 5,000$) go to 2), else continue.

6. Redefine all the cells of "interest" (NSs and SP) as NSs, except from the second to last NS encapsulated for the previous 5,000 time steps which is redefined as a SP. Execute for a second time procedures 2)–5).

Having briefly presented the outline of the model, Eqs. 7 and 8 can be defined to give values for MC and AC of cells constituting sets U, I and N, respectively.

$$MC^t_{(i,j)} = \begin{cases} 0, & \forall i,j : C_{(i,j)} \in U \\ 100, & \forall i,j : C_{(i,j)} \in I \\ 100, & \forall i,j : C_{(i,j)} \in N \text{ and } MC^t_{(i,j)} \geq TMC \end{cases} \tag{7}$$

$$AC^t_{(i,j)} = \begin{cases} 100, & \forall i,j : C_{(i,j)} \in N \text{ and } MC^t_{(i,j)} < TMC \\ 0, & \forall i,j : C_{(i,j)} \in N \text{ and } MC^t_{(i,j)} \geq TMC \end{cases} \tag{8}$$

In order to clarify the model's procedures, every single one will be further explained here. The initialization step includes the definition of parameters that have a great impact on the results of the model. These parameters include the length of the CA grid, the diffusion parameters for the mass concentration ($MCP1$, $MCP2$) and attractants concentration ($ACP1$, $ACP2$), the minimum percentage of attractants concentration detected by the plasmodium, the attraction of the slime mould by attractants (PA – Physarum Attraction) and the threshold of mass concentration that encapsulates a NS (TMC). Also the topology of the NSs and the SP is introduced to the model.

After the initialization and for 50 time steps, diffusion equations are used to calculate the values for AC and MC for every cell in the grid. Every cell uses the values of its neighbours at time step t to calculate the value of the parameters for time step $t + 1$. The contribution to the diffusion of the mass concentration of the von Neumann neighbours ($MCvNN$) of the $C_{(i,j)}$ cell is described in Eq. 9, where k and l are set to values that correspond to the exclusively von Neumann neighbours. Moreover, the contribution to the diffusion of mass concentration of the exclusively Moore neighbours ($MCeMN$) of the $C_{(i,j)}$ cell is described in Eq. 10, where k and l are set to values that correspond to the exclusively Moore neighbours. The total mass concentration for cell $C_{(i,j)}$ for time $t + 1$, is a sum of the contributions of its neighbours with appropriate weights and is described by Eq. 11, respectively.

$$\begin{aligned} MCvNN^t_{(i,j)} = \sum_{k,l} [(1 + PA^t_{(i,j),(k,l)}) \times (1 + IC_{(i,j),(k,l)}) \times \\ MC_{(k,l)} - RS_{(k,l)} \times MC^t_{(i,j)}] \end{aligned} \tag{9}$$

$$\begin{aligned} MCeMN^t_{(i,j)} = \sum_{k,l} [(1 + PA^t_{(i,j),(k,l)}) \times (1 + IC_{(i,j),(k,l)}) \times \\ MC_{(k,l)} - RS_{(k,l)} \times MC^t_{(i,j)}] \end{aligned} \tag{10}$$

$$\begin{aligned} MC^{t+1}_{(i,j)} = & MC^t_{(i,j)} + MCP1 \times \\ & \left[MCvNN^t_{(i,j)} + MCP2 \times MCeMN^t_{(i,j)}\right] \end{aligned} \tag{11}$$

It should be noted that if a neighbouring cell is representing unavailable area, there is no contribution to the diffusion (neither positive nor negative). Moreover, the parameter $IC_{(i,j),(k,l)}$, that takes values as in Eq. 12, includes the gravisensitive characteristic of slime mould, which is depended on the difference of altitudes between two adjacent cells and a coefficient ($InCoef$). That coefficient is used in order to depict different levels of mobility capabilities of the plasmodium. The parameter $PA_{(i,j),(k,l)}$ represents the attraction of slime mould in cell $C_{(i,j)}$ towards the direction of an adjacent cell $C_{(k,l)}$, modeling the attraction towards the higher gradient of attractants. It is equal to a predefined constant (PAP) for the neighbour with the higher value of attractants concentration and equals to the negative value of the same predefined constant for the neighbour across the neighbour with the higher value of attractant. For all the other neighbours the parameter PA is equal to zero. The definition of the PA parameter for cell $C_{(i,j)}$ towards its north neighbour ($C_{(i-1,j)}$) is illustrated in Eq. 13.

$$IC_{(i,j),(k,l)} = (ALT_{(k,l)} - ALT_{(i,j)}) \times InCoef \tag{12}$$

$$PA^t_{(i,j),(i-1,j)} = \begin{cases} PAP, & \text{if } AC_{(i-1,j)} = MAX(AC_{(k,l)} \\ & \forall k,l: \; i-1 \leq k \leq i+1 \\ & \text{and } j-1 \leq l \leq j+1) \\ -PAP, & \text{if } AC_{(i+1,j)} = MAX(AC_{(k,l)} \\ & \forall k,l: \; i-1 \leq k \leq i+1 \\ & \text{and } j-1 \leq l \leq j+1) \\ 0, & \text{else.} \end{cases} \tag{13}$$

Furthermore, the contribution to the diffusion of the attractants of the von Neumann neighbours ($ACvNN$) of the $C_{(i,j)}$ cell is described in Eq. 14. Also, the contribution to the diffusion of the attractants of the exclusively Moore neighbours ($ACeMN$) of the $C_{(i,j)}$ cell is described in Eq. 15. As a result, the total attractants concentration parameter for a $C_{(i,j)}$ cell for time $t+1$ is described in Eq. 16.

$$\begin{aligned} ACvNN^t_{(i,j)} = &(AC^t_{(i-1,j)}) - RS_{(i-1,j)} \times AC^t_{(i,j)} \\ &+(AC^t_{(i,j-1)}) - RS_{(i,j-1)} \times AC^t_{(i,j)} \\ &+(AC^t_{(i+1,j)}) - RS_{(i+1,j)} \times AC^t_{(i,j)} \\ &+(AC^t_{(i,j+1)}) - RS_{(i,j+1)} \times AC^t_{(i,j)} \end{aligned} \tag{14}$$

$$\begin{aligned} ACeMN^t_{(i,j)} = &(AC^t_{(i-1,j-1)}) - RS_{(i-1,j-1)} \times AC^t_{(i,j)} \\ &+ (AC^t_{(i+1,j-1)}) - RS_{(i+1,j-1)} \times AC^t_{(i,j)} \\ &+ (AC^t_{(i-1,j+1)}) - RS_{(i-1,j+1)} \times AC^t_{(i,j)} \\ &+ (AC^t_{(i+1,j+1)}) - RS_{(i+1,j+1)} \times AC^t_{(i,j)} \end{aligned} \tag{15}$$

$$\begin{aligned} AC^{t+1}_{(i,j)} = &CON \times (AC^t_{(i,j)} + ACP1 \times \\ &[ACvNN^t_{(i,j)} + ACP2 \times ACeMN^t_{(i,j)}]). \end{aligned} \tag{16}$$

Note here that if a neighbouring cell is representing unavailable area, there is no contribution to the diffusion (neither positive nor negative), as in the diffusion of mass concentration. Moreover, the multiplication with the parameter CON, provides the imitation of the consumption of the attractants substances by the plasmodium.

After every 50 time steps of calculating the diffusion equations in the available area, if any NS is covered with the predefined MC (TMC), it is connected with a SP with a path that follows the gradient of the MC to the higher value. More specifically, starting from the cell representing the encapsulated NS, the adjacent cell with the higher MC value is selected to participate to the tubular network. Then the cell selected to participate to the tubular network selects the next cell from its neighbours with the higher MC value to participate to the tubular network and so on, until a SP is reached.

Then, this NS will be transformed to a SP ($MC = 100$) and will act as a SP for the remaining time steps, as illustrated in Eqs. (7) and (8), respectively. If more NSs are covered with the predefined MC, they are connected to the nearest SP and they are all transformed to SPs.

Experiments-Results

To study the successful mimicking of the slime mould's behaviour by the model, some basic graphical configurations have been used as input data. Namely, a rectangular CA grid is assumed with the surrounding cells set as unreachable cells (in set U), one cell is defined as a SP which is located on the lower right region of the surface, while one cell is defined as a NS (or a destination point) which is located on the upper left region of the surface. These assumptions are the same for all the paradigms presented here. Nonetheless, the parameters for the model are same throughout all the experiments and illustrated in Table 1. On the other hand, the elevations are different in all the experiments to study their effect on the network produced by the model.

Table 1: Parameters' values for the CA model.

Parameter	Value	Parameter	Value
$MCP1$	0.08	$GridLength$	60×60
$MCP2$	0.01	CON	0.95
$ACP1$	0.05	PA	0.7
$ACP2$	0.01	TMC	0.2
$InCoef$	0.05		

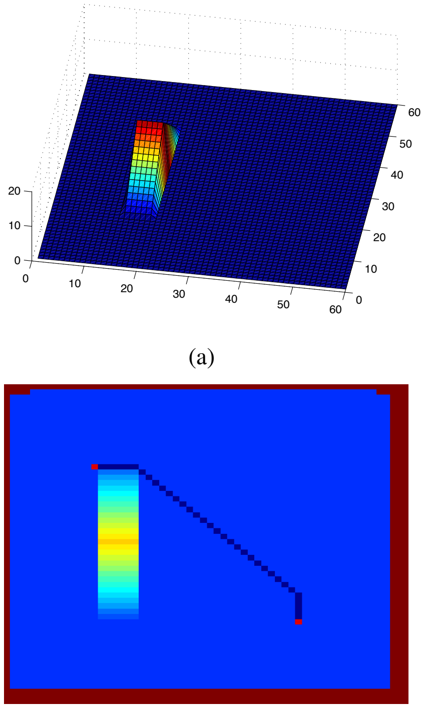
The following illustrations (Figs. 1(b) - 5(b)) depict the results of the model after 2000 time steps. The colors of every cell in that figures are encoded as follows. Light brown colored cells represent the unreachable experimental surface. Light blue cells correspond to reachable surface without any elevations. Red cells depict points of interest (SP or NS). Dark blue cells indicate the tubular network designed by the model. Finally, the location of the obstacle is indicated with light blue to orange colored cells that encode their altitude from a lower to a higher level, respectively.

In the first experiment, an obstacle is placed near the destination point as depicted in Fig. 1(a). The purpose of this experiment is to test whether the model will route around the obstacle that is formed as a double ramp which intersects the theoretical shortest path. After 2000 time steps the resultant connection is definitely avoiding the obstacle, but also keeping the total length as short as possible as realized in Fig. 1(b).

Another experiment is conducted with an obstacle placed in the center of the experimental surface, as illustrated in Fig. 2(a). Once more, the obstacle is located in the path of the shortest connection between the two points and has a shape similar to a pyramid. Thus, the model designs a connection that bypasses the obstacle from the lower segment of the surface, as shown in Fig. 2(b).

In the next experiment the same pyramid-shaped obstacle is placed a bit lower than the center of the surface (Fig. 3(a)). In that way the model is biased to provide a connection routed in the upper section of the surface. That is the case here, as presented in Fig. 3(b).

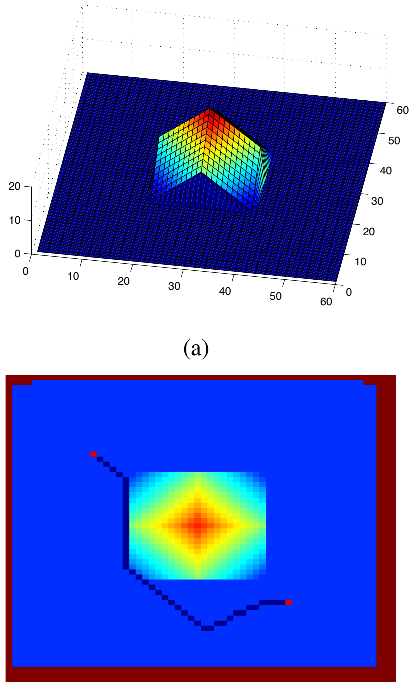
In order to study whether the provided connection may overcome an obstacle, an experiment with a larger than the previous, pyramid-like obstacle placed in the center of the surface is conducted (Fig. 4(a)). Furthermore, the obstacle has a larger base but its higher altitude is lower than the previous obstacles, namely the altitude of the top of the pyramid is 5 units. As illustrated in Fig. 4(b) the model has evalu-



(a)

(b)

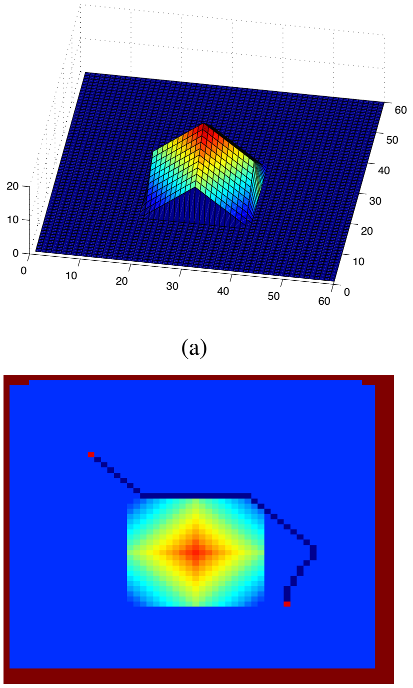
Figure 1: (a) 3D illustration of the experimental terrain with an obstacle placed near the destination. (b) Results of the model with the given elevations.



(a)

(b)

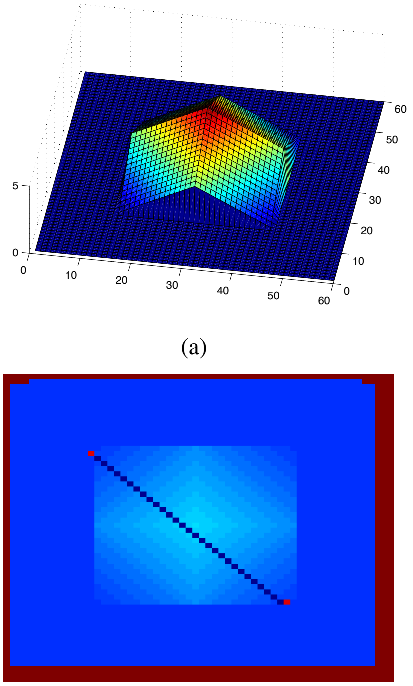
Figure 2: (a) 3D illustration of the experimental terrain with a pyramid-like obstacle placed in the center of the terrain. (b) Results of the model with the given elevations.



(a)

(b)

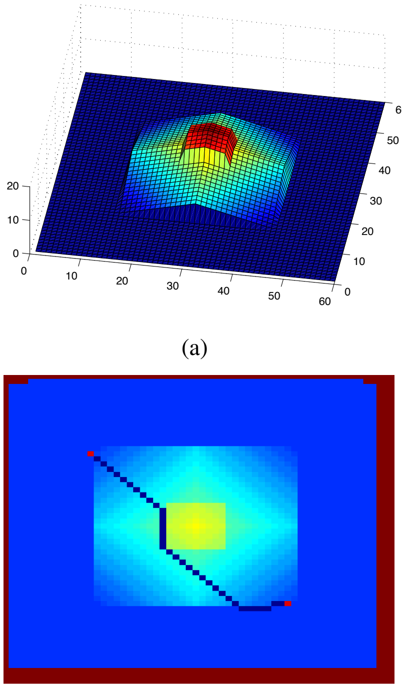
Figure 3: (a) 3D illustration of the experimental terrain with a pyramid-like obstacle placed a bit lower than the center of the terrain. (b) Results of the model with the given elevations.



(a)

(b)

Figure 4: (a) 3D illustration of the experimental terrain with a pyramid-like obstacle with higher point at altitude 5 units placed in the center of the terrain. (b) Results of the model with the given elevations.



(a)

(b)

Figure 5: (a) 3D illustration of the experimental terrain with an obstacle with higher point at altitude 20 units and a inhomogeneous increase in its elevation, placed in the center of the terrain. (b) Results of the model with the given elevations.

ated the route bypassing the obstacle as less efficient than the straight line overcoming the obstacle. As a result, the straight line path is the one that was finally drawn. Moreover, note here that the base of the obstacle is adjacent to the two points, a fact that makes the bypassing route the longest possible.

Finally, for the last experiment, an obstacle with two levels, introducing some inhomogeneity in the altitude of the surface, is placed as shown in Fig. 5(a). The base of the obstacle here is the same as the previous experiment, however, there is a steeper increase of the altitude in the center of the obstacle. That inhomogeneity causes the resultant path to maneuver around the higher altitude as depicted in Fig. 5(b).

Note here that in the cases of Figs. 2, 3 and 5, the model has connected the two points through a fairly short path, however, it did not follow the straight lines on the shortest possible path. That is partially based on the fact that the cells placed on the boundaries of the CA grid have no contributions, positive or negative, to the diffusion equations, as described previously. As a result, the concentration of the simulated protoplasmic mass will be accumulated faster on the barriers of unreachable and reachable surface. In addition to that, the grid length is defined at a marginal level for demonstration reasons. As a result, the approximation of the diffusion equations on crucial cells near the shortest possible path, are greatly affected from the boundaries. Having explained that, the tubular network produced by the model has slight deviations towards the boundaries of the CA grid, where some of the simulated protoplasmic mass concentration is higher than expected. Nonetheless, the laboratory experiments with slime mould do not produce networks consisted of straight edges, thus, this can not be considered as a disadvantage of the model.

Conclusions

The plasmodium of *Physarum polycephalum* has been used as a biological substrate for analogous/biological computers. The inputs on this new kind of computer were often the topology of points of interest (SP and NSs) and the outputs were the tubular network developed by the plasmodium. Lately, the inputs used have been made more complex, introducing the gravisensitive and photo-phobic characteristics of slime mould. Thus, the software based models developed in previous studies were redesigned, to accommodate the gravisensitive factor in the network development procedure. The updated model was, also, based on CA principles due to their inherent parallel nature and lack of central control, in accordance with the biological substrate. The resulting networks provided by the initial experiments presented in this paper are in good agreement with the behaviour of the real slime mould, avoiding steep obstacles while keeping the network as efficient as possible in terms of total length.

As an aspect of future work, the applicability of the model will be tested in experiments conducted with more complicated data. Namely, data based on the morphological characteristics of terrains of countries will be used and the results of the model will be compared with the results of manmade motorways and the results of laboratory experiments with slime mould on 3D terrains. Finally, the photo-phobic characteristic of slime mould will be included in the biomimicking model, to obtain even more realistic results.

References

Adamatzky, A. (1994). *Identification Of Cellular Automata*. Taylor & Francis Group.

Adamatzky, A. (2010). *Physarum Machines: Computers from Slime Mould*. World Scientific series on nonlinear science. World Scientific.

Adamatzky, A. (2013). Bio-Imitation of Mexican Migration Routes to the USA with Slime Mould on 3D Terrains. *Journal of Bionic Engineering*, 10:242–250.

Adamatzky, A. (2014). Route 20, Autobahn 7, and Slime Mold: Approximating the Longest Roads in USA and Germany With Slime Mold on 3-D Terrains. *IEEE Transaction on Cybernetics*, 44(1):126–137.

Adamatzky, A. and Jones, J. (2015). On using compressibility to detect when slime mould completed computation. *Complexity*.

Ashlock, D. and McNicholas, S. (2013). Fitness Landscapes of Evolved Apoptotic Cellular Automata. *IEEE Transactions on Evolutionary Computation*, 17(2):198–212.

Block, I., Briegleb, W., and Wohlfarth-Bottermann, K. E. (1986). Gravisensitivity of the acellular slime mold Physarum polycephalum demonstrated on the fast-rotating clinostat. *European Journal of Cell Biology*, 41:44–50.

Block, I., Rabien, H., and Ivanova, K. (1998). Involvement of the second messenger cAMP in the gravity-signal transduction in Physarum . *Advances in Space Research*, 21:1311–1314.

Dourvas, N., Tsompanas, M.-A., Tsalidis, P., and Sirakoulis, G. C. (2015). Hardware acceleration of Cellular Automata Physarum Polycephalum model. *Parallel Processing Letters*, 25(01).

Feynman, R. P. (1982). Simulating physics with computers. *International Journal of Theoretical Physic*, 21:467–488.

Georgoudas, I. G., Kyriakos, P., Sirakoulis, G. C., and Andreadis, I. T. (2010). An FPGA implemented cellular automaton crowd evacuation model inspired by the electrostatic-induced potential fields. *Microprocessors and Microsystems*, 34(7):285–300.

Glockner, G., Golderer, G., Werner-Felmayer, G., Meyer, S., and Marwan, W. (2008). A first glimpse at the transcriptome of physarum polycephalum. *BMC Genomics*, 9(6).

Jendrsczok, J., Ediger, P., and Hoffmann, R. (2009). A scalable configurable architecture for the massively parallel gca model. *International Journal of Parallel, Emergent and Distributed Systems*, 24(4):275–291.

Mardiris, V., Sirakoulis, G. C., Mizas, C., Karafyllidis, I., and Thanailakis, A. (2008). A CAD system for modeling and simulation of computer networks using cellular automata. *IEEE transactions on systems, man and cybernetics. Part C, Applications and reviews*, 38(2):253–264.

Mayne, R., Tsompanas, M.-A., Sirakoulis, G., and Adamatzky, A. (2015). Towards a slime mould-FPGA interface. *Biomedical Engineering Letters*, in press.

Nakagaki, T., Iima, M., Ueda, T., Nishiura, Y., Saigusa, T., Tero, A., Kobayashi, R., and Showalter, K. (2007). Minimum-risk path finding by an adaptive amoeba network,. *Phys. Rev. Lett*, 99(6):068104–1–0 – 68104–4.

Progias, P. and Sirakoulis, G. C. (2013). An FPGA processor for modelling wildfire spread. *Mathematical and Computer Modeling*, 57(5-6):1436–1452.

Sirakoulis, G. and Adamatzky, A. (2014). *Robots and Lattice Automata*. Emergence, Complexity and Computation. Springer International Publishing.

Sirakoulis, G. C. and Bandini, S., editors (2012). *Lecture Notes in Computer Science*, volume 7495. Springer, Santorini Island, Greece.

Sirakoulis, G. C., Karafyllidis, I., and A., T. (2000). A cellular automaton model for the effect of population movement on epidemic propagation. *Ecological Modelling*, 133(3):209–223.

Sirakoulis, G. C., Karafyllidis, I., and Thanailakis, A. (2003). A CAD system for the construction and VLSI implementation of cellular automata algorithms using VHDL. *Microprocessors and Microsystems*, 27:381–396.

Spezzano, G., Talia, D., Di Gregorio, S., Rongo, R., and Spataro, W. (1996). A parallel cellular tool for interactive modeling and simulation. *Computing in Science and Engineering*, 3:33–43.

Tero, A., Takagi, S., Saigusa, T., Ito, K., Bebber, D., Fricker, M., Yumiki, K., Kobayashi, R., and Nakagaki, T. (2010). Rules for Biologically Inspired Adaptive Network Design. *Science*, 327:126–137.

Tsompanas, M.-A., Sirakoulis, G., and Adamatzky, A. (2014a). Evolving Transport Networks with Cellular Automata Models Inspired by Slime Mould. *IEEE Transactions on Cybernetics*, in press.

Tsompanas, M.-A., Sirakoulis, G., and Adamatzky, A. (2014b). Slime Mould Imitates Greek Motorways: From real organism to silicon. *Natural Computing*, in press.

Tsompanas, M.-A. I. and Sirakoulis, G. C. (2012). Modeling and hardware implementation of an amoeba-like cellular automaton. *Bioinspiration & Biomimetics*, 7:036013.

Ueda, T., Mori, Y., Nakagaki, T., and Kobatake, Y. (1988). Action spectra for superoxide generation and uv and visible light photoavoidance in plasmodia of physarum polycephalum. *Photochemistry and Photobiology*, 48(5):705–709.

Vichniac, G. Y. (1984). Simulating physics with cellular automata . *Physica D: Nonlinear Phenomena*, 10:96–116.

von Neumann, J. (1966). *Theory of Self-reproducing Automata*. University of Illinois Press.

Werfel, J., Mitchell, M., and Crutchfield, J. P. (2000). Resource sharing and coevolution in evolving cellular automata. *IEEE Transactions on Evolutionary Computation*, 4(4):388–393.

Wilding, N., Trew, A., Hawick, K., and Pawley, G. (1991). Scientific modeling with massively parallel simd computers. *Proceedings of the IEEE*, 79(4):574–585.

Wolfram, S. (1986). *Theory and Applications of Cellular Automata*. Singapore: World Scientific.

Behavioral Diversities of Morphogenetic Collective Systems

Hiroki Sayama

Binghamton University, State University of New York, USA
sayama@binghamton.edu

We conducted comparative analyses of self-organized behaviors generated by morphogenetic collective systems of four distinct classes: homogeneous collectives (Class A); heterogeneous collectives (Class B); heterogeneous collectives with dynamic differentiation/re-differentiation (Class C); and heterogeneous collectives with dynamic differentiation/re-differentiation and local information sharing (Class D) (Sayama 2014). In previous work, behaviors of morphogenetic swarm chemistry were sampled by Monte Carlo simulations and characterized in 24 kinetic, topological and dynamical features, although statistical analyses remained simple mean difference tests analyzing each feature independently.

To elucidate potential differences in behavioral diversity between the four classes, here we measured class-level properties of sampled behavior distributions in the 24-dimensional behavioral feature space. The entire sample set (including samples from all four classes) was first standardized and transformed into uncorrelated components by principal component analysis (Fig. 1). Then the following three measurements were calculated in the feature space for each class to characterize its behavioral diversity: (a) approximated volume of behavior coverage (i.e., product of ranges [max – min] of all 24 components), (b) average pairwise distance of behaviors between two randomly selected samples, and (c) differential entropy (Cover & Thomas 1991) of the smoothed probability density function constructed for the first four principal components (which corresponded to 65% of total variance).

Figure 2 summarizes the results. All the measurements showed that classes C and D produced greater behavioral diversities than A and B (counter to the previous observation that properties of C and D were generally between A and B). This suggests that dynamic differentiation/re-differentiation present in C and D contributed to the production of more diverse behaviors in morphogenetic collective systems.

References

Cover, T. M., Thomas, J. A. (1991). *Elements of Information Theory*. John Wiley & Sons, Hoboken, NJ. Chapter 9.

Sayama, H. (2014). Four classes of morphogenetic collective systems. In *Artificial Life 14*, pages 320–327. MIT Press, Cambridge, MA.

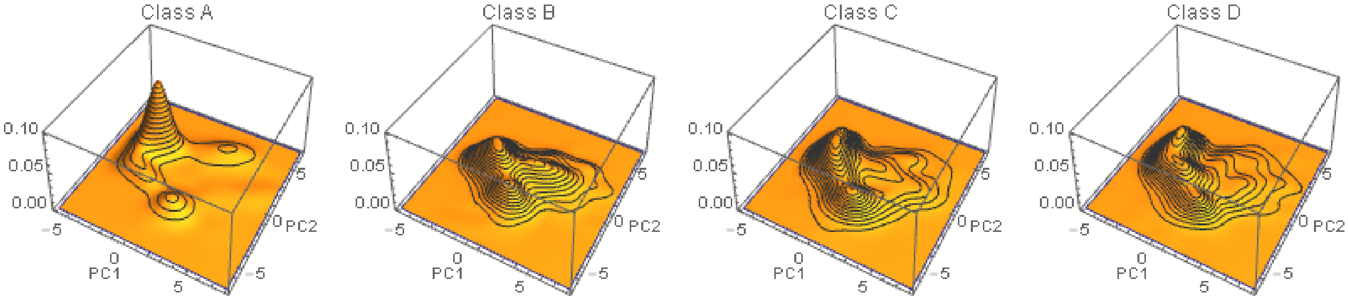


Figure 1. Probability density functions of swarm behaviors, plotted over a 2D space made of the first two principal components.

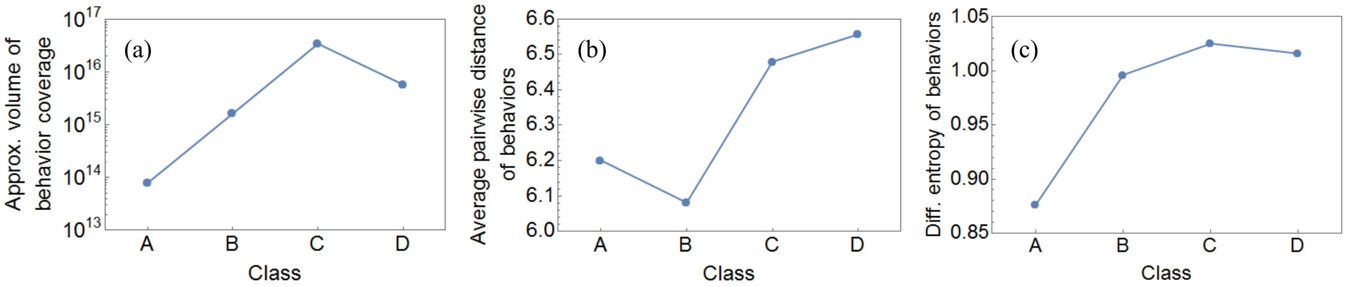


Figure 2. Comparison of behavioral diversity between four classes. (a) Approximate volume of behavior coverage. (b) Average pairwise distance of behaviors. (c) Differential entropy of behaviors. C & D consistently showed greater diversities than A & B.

Morphogenesis and Replication of Multi-Cellular Organisms with Evolved Variable Length Self-Modifying Genomes

Stefano Nichele[1] and Gunnar Tufte[1]
[1]Norwegian University of Science and Technology, Trondheim, Norway
{nichele, gunnart}@idi.ntnu.no

Abstract

The genomes of biological organisms are not fixed in size. They evolved and diverged into different species acquiring new genes, thus having different lengths. In a way, biological genomes are the result of a self-assembly process where parent's genes sometimes are not only copied to offspring but are also duplicated. There is scientific evidence that all species have evolved and diverged from a common ancestor, i.e. last universal ancestor (LUA), and the mechanisms of gene duplication played an important role for genetic novelty and evolutionary innovation. This complexification process is a plausible explanation of how efficient and robust genomes have evolved. Morphogenesis is a result of the inherent scalability of biological genomes. In the artificial domain, evolutionary morphogenetic systems often have static size genomes, e.g. chosen beforehand by the system designer by trial and error or estimated a priori with complicated heuristics. As such, the maximum evolvable complexity is predetermined. This is in contrast with open-ended evolution in nature.

Previous work (Nichele and Tufte, 2014) has shown that artificial genomes may also grow in size during evolution to produce high-dimensional solutions incrementally. The proposed evolutionary growth of genome representations has been investigated for artificial cellular organisms with indirect encodings. In practice, genomes start with a single gene and acquire new genes when necessary, thus increasing the degrees of freedom and expanding the available search-space. Cellular Automata (CA) have been used as test bed for two different problems: morphogenesis and replication. It has been shown that CA instruction-based development (Bidlo and Skarvada, 2008) allows evolutionary growth of genomes, providing more compact and effective genomes than traditional CA table-based genomes, without the need of specifying all the neighborhood regulatory combinations (Nichele, 2015). An example of morphogenesis of French flag[1] pattern is shown in Figure 1, where a genotype made of 14 instructions was evolved incrementally. For details on genotype instructions see (Nichele and Tufte, 2014). Another example is shown in Figure 2, where the same structure self-replicates.

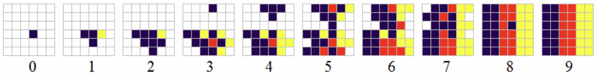


Figure 1: Morphogenesis of French flag[1] with evolutionary growth of genomes and instruction-based development.

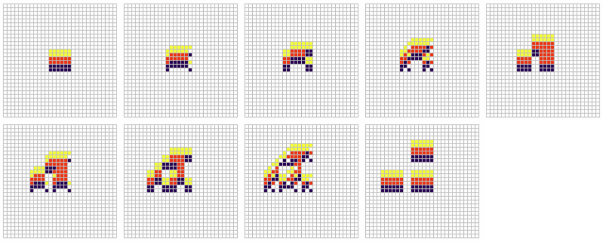


Figure 2: Self-replication of French flag[1] pattern. Each replica is also a replicator.

Future work will address the following: 1) Let the growth happen both in terms of genotype size, i.e. number of genes, and number of states that each cell can hold. This may help to achieve true complexification, meaning that the boundaries would not be fixed by the state space, which is more or less fixed in every artificial system. 2) Allow genotype instructions that can modify the genotype itself, e.g. as in self-modifying cartesian genetic programming (SMCGP). This may allow the diversification of cell programs (as genotype activation/regulation mechanism), enabling potentially hierarchical organization and aggregation of cells, similar to biological cells, tissues, organs and organisms. This may encourage the emergence of some sort of artificial stem cells mechanisms (stem cells are replicators by definition) that may allow morphogenesis as in Figure 1 followed by replication as in Figure 2, both in the same process. We argue that an appropriate challenge for morphogenetic systems would be to target evolution of complex morphologies and structures, potentially at natural levels of complexity.

References

Bidlo, M. and Skarvada, J. (2008). Instruction-based development: From evolution to generic structures of digital circuits. *KES Journal*, 12(3):221–236.

Nichele, S. (2015). Evolvability, complexity and scalability of cellular evolutionary and developmental systems. Norwegian University of Science and Technology, PhD Thesis.

Nichele, S. and Tufte, G. (2014). Evolutionary growth of genomes for the development and replication of multicellular organisms with indirect encoding. In *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pages 141–148.

[1] Here different colors are used for print purposes.

Modelling the Role of Trail Pheromone in the Collective Construction of Termite Royal Chambers

Nicholas Hill[1] and Seth Bullock

Institute for Complex Systems Simulation, University of Southampton, UK, SO17 1BJ

[1]n.c.hill@soton.ac.uk

Abstract

Experiments with worker termites constructing a royal chamber around a termite queen in species *Macrotermes subhyalinus (Rambur)* have shown that both trail and cement pheromones are involved and necessary for the successful formation of pillars during the building process. However, earlier models of the construction were able to demonstrate stigmergic pillar formation with cement pheromone alone. We present results from a new three-dimensional agent-based model, developed to investigate the role of trail pheromone in the construction process. The model is able to demonstrate how, if the properties of the cement pheromone are altered so that its attractive influence is more localised than in earlier models, termites are unable to produce significant pillar formation. The model shows how the addition of trail deposition and following effectively increases the range of the stigmergic effect so that pillar formation is restored. The presence of trail pheromone also results in pillars which are narrower than those produced by cement pheromone alone, and which show more pronounced lateral extensions. Additionally the paths that the termites take from the termite queen to building sites become more directed with time. These features are in keeping with observation and have not been previously modelled.

Introduction

As a scientific field, artificial life has studied the natural processes through which collections of simple components can give rise to complex systemic behaviour. One example of such a collective process is *morphogenesis*, the development of coherent structure in biological systems. Colonies of social insects demonstrate this process when, for example, a collective effort is required in order to construct a nest or when the individuals in a colony coalesce to form useful structures, such as rafts (Wilson, 1971).

Morphogenetic engineering is the study of the ways in which artificial morphogenetic systems can be created and directed to ensure the formation of structures with pre-specified requirements (Doursat et al., 2012). Understanding the dynamics of social insect construction may enable the extraction of design principles which facilitate the creation of teams of agents that are capable of building functional structures in remote and hostile environments, in an environmentally adaptive, self-organised and self repairing manner.

Termites and Collective Construction

Some species of termite are able to construct nests in the form of mounds that are orders of magnitude greater in size than the individual termites and which have several functionally distinct sections (Howse, 1970; Wilson, 1971). There is no evidence, however, that these termites rely on centralised blueprints or engage in planning, deliberation or negotiation during mound construction. Instead they act independently and instinctively, responding to multiple local cues in their immediate environment, including air flow, humidity and temperature, and also to the presence of any existing built structures (Grassé, 1959; Stuart, 1967; Howse, 1970). The ways in which combinations of environmental signals and instinctive responses enable termites to self-organise their building effort are not well understood. There is evidence, however, that the trail pheromone that they synthesize and deposit acts as a chemical signal diffusing through the environment and is a key factor in many aspects of termite behaviour (Stuart, 1970; Leuthold et al., 1976). Accordingly, here we develop a model of an early stage of termite nest construction in order to investigate how trail pheromone may be of central importance.

In species *Macrotermes subhyalinus (Rambur)*, worker termites begin nest construction by building a dome-like structure, or royal chamber, around their much larger and immobile queen (Bruinsma, 1979). Construction of a royal chamber is one aspect of the nest building process that is relatively reproducible and therefore Bruinsma (1979) designed a series of controlled experiments in order to investigate the ways in which pheromones and other factors are involved in the coordination of the construction process.

Bruinsma (1979) observed royal chamber construction after the introduction of a number of worker termites into a perspex box containing a single new cessile queen. Initially, the termites spend time grooming the abdomen of the queen before individuals move away from the queen, picking up a pellet of soil and tracing a meandering path out to a distance of about 2-5cm where the pellet is placed and secured with faecal cement before returning to the termite queen to groom again. Gradually more and more of the termites repeat this process and in this way a behavioural cycle develops.

Initially, the distribution of deposited soil pellets around the queen appears random but aggregations of material quickly form as termites begin to preferentially approach the existing building sites and add material to them. The distribution of built material becomes concentrated in a few locations where pillars form. Initially the paths taken by a termite before placing a pellet are serpentine and relatively long but, over time, these paths become shorter and straighter, running more directly from the termite queen to a pillar. When the pillars reach a certain height, termites may begin to build horizontally outwards to form lateral extensions termed *lamellae*. Lamellae from adjacent pillars may eventually join to form arches.

Bruinsma (1979) deduced that the queen termite emits a *building pheromone* which the worker termites spread over the abdomen while grooming. By diffusing away from the queen's body, this pheromone acts as a template for the building process, effectively defining a *deposition zone* around the queen by inducing the termites to place soil pellets mainly at a distance where the level of pheromone is within a specific range. Bruinsma (1979) also found that the faecal cement which the termites use to fix soil pellets contains *cement pheromone*. This pheromone both attracts termites and makes them more likely to deposit a soil pellet. Earlier observations of termite construction by Grassé (1959) led him to theorise that an attractive pheromone infused into the material that termites build with was responsible for the positive feedback effect that results in aggregation of built material and then pillar formation. Grassé (1959) called this process *stigmergy*.

A two-dimensional mathematical model by Deneubourg (1977) and a two-dimensional cellular automaton by Courtois and Heymans (1991) demonstrated how this positive feedback can occur in a roughly homogeneous distribution of pheromone-infused building material, when material is preferentially moved towards areas with higher levels of the pheromone. Any initial small fluctuations in the distribution of material are amplified resulting in a more peaked distribution. It was hypothesised that these peaks could be identified with the early stages of pillar formation. An extension of the mathematical model by Bonabeau et al. (1998) to include features of Bruinsma's royal chamber experiments showed that the introduction of a pheromone template emitted by the termite queen can restrict the formation of peaks such that they only occur within a certain range of her.

A three-dimensional agent-based version of the extended mathematical model (Ladley and Bullock, 2004, 2005), designed to include more realistic physical constraints on termite movement and building, also showed that cement pheromone could act as a recruiting mechanism to enable the formation of pillars in a deposition zone around an artificial termite queen. Unlike the previous two-dimensional models, in this model it was possible for termites to build truly three-dimensional structures, implying the logical possibility of lamellae construction. However, the model was not able to demonstrate lamellae arising as a result of cement-pheromone-mediated positive feedback in the building process.

Further findings by Bruinsma (1979) suggest that the attractive nature of cement pheromone cannot be the only mechanism responsible for pillar formation. The distance over which cement pheromone influences the actions of worker termites was experimentally determined to be approximately 1-1.5cm, which is less than the distance between the queen and the deposition zone. In addition, experiments revealed that *trail pheromone* was present in the soil between the queen termite and the deposition zone. Although no specific trail patterns could be identified, Bruinsma (1979) showed that when the termites were prevented from being able to deposit trail pheromone, they were unable to complete construction of the royal chamber and instead built flattened ridges closer to the termite queen, with no evidence of pillar formation. Finally, the observation by Bruinsma (1979) that the paths taken by the worker termites from the queen to the deposition zone become less serpentine as the construction of the royal chamber progresses suggests that they may be leaving trails of pheromone which are influencing the paths taken by other termites and enabling them to more quickly reach active building sites.

To date, no model has incorporated the ability of the termites to deposit and follow trail pheromone while building. Consequently, here, we extend the three-dimensional agent-based model of Ladley and Bullock (2004, 2005) in order to investigate the role of trail pheromone in the building process. Our intention is to explore whether the inclusion of trail pheromone can lead to the demonstration of the following observed features of royal chamber construction that have not been modelled to date:

1. Stigmergic recruitment by trail pheromone leading to pillar formation,
2. Characteristic failure of the build process in the absence of trail laying,
3. Formation of lamellae,
4. The appearance of increasingly direct paths from the termite queen to the deposition zone.

In the following sections we first describe the design of the model, before presenting simulation results. We then provide a discussion before the paper concludes.

Methods

In order to model trail laying and following it is necessary to introduce a simple approximation of the termite behavioural cycle observed by Bruinsma so that the termites have distinct behavioural modes during which they are either grooming the queen termite, travelling to the deposition zone, or returning to the queen after building. Depending on model parameters, when en route to the deposition zone or when deciding whether or not to place a piece of building material, termites may be influenced by some combination of the local concentrations of trail pheromone and/or cement pheromone. When returning to the queen after building, termites use the building pheromone in order to orient towards the queen, and lay trail pheromone as they move.

Simulations were carried out using a three-dimensional agent-based model based on that of Ladley and Bullock (2004, 2005) in order to incorporate the same logistic constraints and so that comparisons can be made with the output of that model. In the model, space is represented by a three-dimensional cubic lattice of size $X \times Y \times Z$, where Z labels the extent of the lattice in the vertical dimension. All locations in the world except edge locations therefore have a set of 26 adjacent *neighbours*, of which six are *cardinal neighbours*; two locations are cardinal neighbours if they share a face.

Each location may contain:

1. A single block of one of three types, representing pellets of building material, sections of ground and sections of termite queen. Locations not containing a block are deemed *empty*.

2. Zero or more worker termites. The total number of termites in the model is denoted n and each follows the behavioural cycle outlined in figure 1. Termites cannot share locations with blocks.

3. Concentration values for three types of pheromone; building pheromone, cement pheromone and trail pheromone.

Simulations are synchronous and operate for a fixed number of discrete time steps, t_{max}. At each time step, the model updates the behavioural mode and location of each worker termite, adds newly placed blocks of building material as a result of building activity, and updates the concentration values of pheromones at each location.

The locations of blocks of ground and termite queen are fixed at the start of each simulation. Ground blocks are placed in all locations for which $z = 1$, and the termite queen is represented as a predetermined configuration of queen blocks placed on top of the ground and centrally with respect to x and y. All other locations are initially empty. The initial locations of the worker termites are randomly selected from the set of all empty locations that are cardinal neighbours of locations containing queen blocks. All pheromone values are initially set to 0.

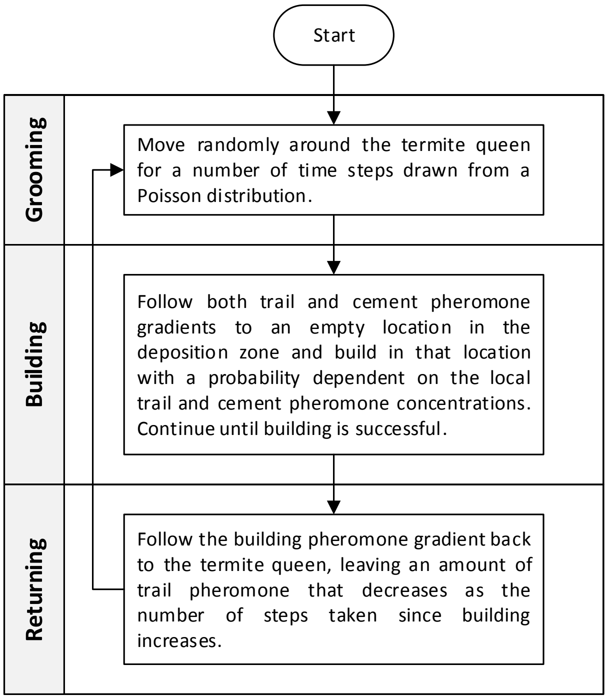

Start
Grooming
Move randomly around the termite queen for a number of time steps drawn from a Poisson distribution.
Building
Follow both trail and cement pheromone gradients to an empty location in the deposition zone and build in that location with a probability dependent on the local trail and cement pheromone concentrations. Continue until building is successful.
Returning
Follow the building pheromone gradient back to the termite queen, leaving an amount of trail pheromone that decreases as the number of steps taken since building increases.

Figure 1: The termite behavioural cycle.

Pheromones

Each of the pheromone types in the model has a distribution represented by a set of real values defined across all locations. The distributions are parameterised independently and subject to processes of diffusion and evaporation which are applied on every 5^{th} time step.

Pheromone values in all non-edge locations are subject to diffusion between cardinal neighbours. Diffusion between two cardinal neighbours is proportional to the difference in the pheromone values at the two locations and is implemented using the finite volume method (Hirsch, 1988). The amount of pheromone that diffuses from one location with a volume of pheromone U_1, to a cardinal neighbour with a volume of pheromone U_2, is given by $\Delta U = -\rho(U_1 - U_2)$, where ρ is the *diffusion coefficient*. Pheromone cannot diffuse into locations containing blocks, with the proportion that would have done so remaining in the original location. Pheromone can diffuse into edge locations but is then removed from the simulation so that it cannot diffuse back into non-edge locations. Evaporation of each pheromone type is modelled by multiplying the pheromone value of that type in each location by an evaporation constant, $0 < \nu \leq 1$, at each update.

A constant volume of building pheromone is emitted by the termite queen and this is represented by setting the building pheromone value in all locations containing queen blocks to a constant value ϕ_Q at each time step. Blocks representing pellets of building material contain an initial amount of cement pheromone ϕ_C which is set at the location and time that a block is placed by a termite. The level of cement pheromone remaining at the site of block placement is multiplied at each time step by a factor $0 \leq r \leq 1$, representing the fraction lost through denaturing. Trail pheromone is deposited by termites in each location that they move through on their way back to the queen after placing a pellet of building material (see below).

Termites

At each time step a termite in location l can in principle move to all locations in the set, L, of 26 neighbours surrounding l. However, in practice a termite can move only to a subset of these locations; the *target set*, K. The locations that a termite cannot move to are represented by the sets:

1. E: all edge locations,
2. F: all locations containing a block,
3. A: all locations with no cardinal neighbour containing a block (locations deemed to be in mid air),
4. W: empty non-cardinal neighbour locations within L for which all cardinal neighbours that share a side or edge with location l contain a block (these locations are effectively on the other side of a wall),
5. D: a set of the previous locations that a termite has moved through (this set is designed to make trail gradient following more polarised by preventing termites that are following a trail from immediately turning around and walking back in the opposite direction).

Additionally, when in grooming mode termites are subject to a restriction set R which contains all locations that are not cardinal neighbours of queen blocks. When not grooming, $R = \emptyset$. The target set K of locations a termite can move to is therefore given by:

$$K = L \setminus (E \cup F \cup A \cup R \cup W \cup D) \tag{1}$$

Once K has been determined, a target location to move to, $k \in K$, is computed depending on the behaviour of the termite. If moving randomly then the probability of moving to each target location $k \in K$ is identical and given by $P_m = 1/|K|$. If following a single pheromone type then the probability is given by:

$$P_m(k,l) = \frac{\Delta U(k,l) + \mu}{\sum_k \Delta U(k,l) + |K|\mu} \tag{2}$$

Here, $\Delta U(k,l)$ is any positive difference in pheromone values between k and l, given by:

$$\Delta U(k,l) = \begin{cases} U(k) - U(l), & \text{if } (U(k) - U(l)) \geq 0 \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

and μ is a sensitivity parameter which represents the extent to which a termite will move randomly rather than move towards a location with a higher pheromone value. This parameter biases the selection of a target location such that as $\sum_k \Delta U(k,l) \to 0$, $P_m(k,l) \to 1/|K|$ and is independently defined for each pheromone type. This ensures that if pheromone differences are large relative to μ then a termite will be more likely to be influenced by those differences, whereas if they are small then a termite will be more likely to move randomly. Once $P_m(k,l)$ has been determined for all $k \in K$, a roulette wheel selection algorithm is used to choose k.

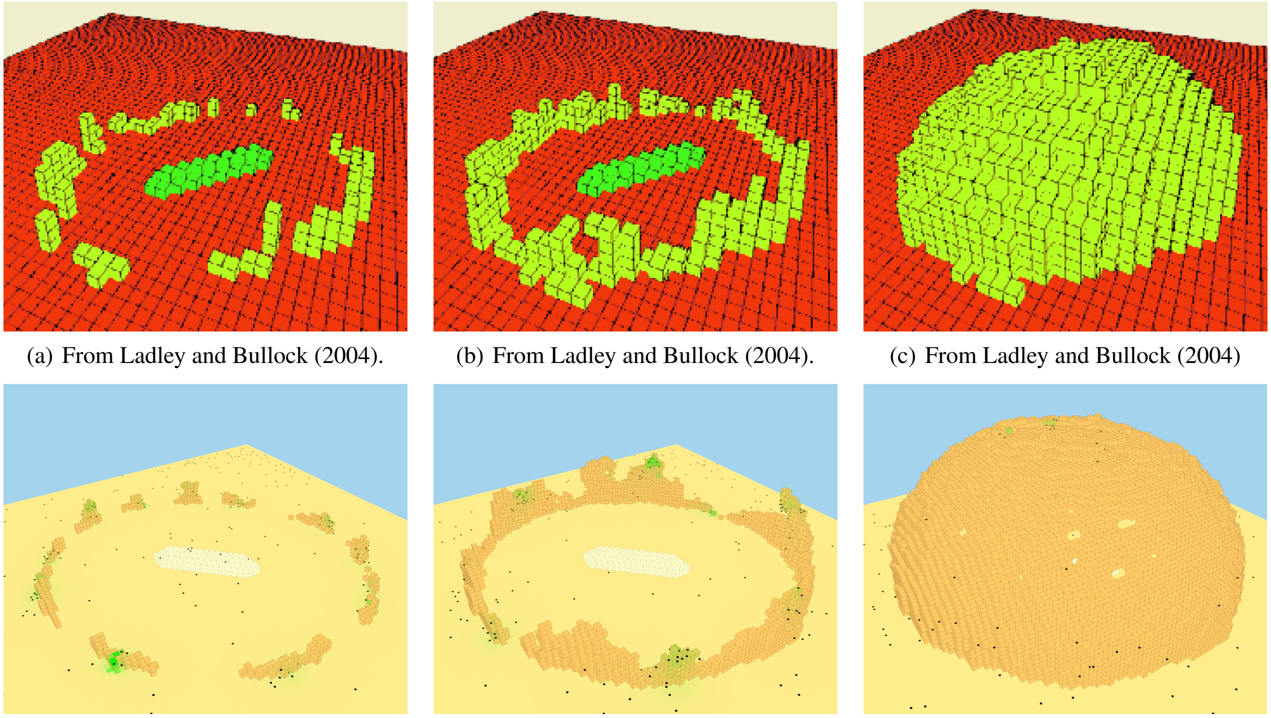
If more than one type of pheromone is influencing the movement of a termite then the weighted sum $(1 - w)P_m^1(k,l) + wP_m^2(k,l)$ is used in the roulette wheel selection algorithm to represent the relative extent to which it is influenced by each type of pheromone ($0 \leq w \leq 1$).

Grooming Termites begin a simulation by grooming the queen termite. During this time they move randomly around the set of locations that are cardinal neighbours of queen blocks. The time that each termite spends grooming is randomly drawn from a Poisson distribution of expected value λ. After this time has expired, a termite switches to building behaviour mode. Grooming behaviour ensures that the termites leave the queen to build at different times and from different locations adjacent to the queen.

Building In building behaviour mode it is assumed that each termite has a pellet of building material. The termites search for a location to place the pellet by following both cement and trail pheromone gradients until they reach a location where they could potentially build. This deposition zone defines the overall shape of the royal chamber and is defined as all locations l for which $Q_{min} \leq Q(l) \leq Q_{max}$, where $Q(l)$ is the level of building pheromone at location l, and Q_{min} and Q_{max} are model parameters. Once a termite arrives at a location l in this zone it may place a block of material if at least one of the following is true:

- The cardinal neighbour above l or the cardinal neighbour below l contains a block,
- One of the horizontal cardinal neighbours of l contains a block which has a block either directly above or below it,
- The location l is a horizontal cardinal neighbour of one or two of a set of three neighbouring locations that each contain blocks and which are horizontal cardinal neighbours of each other (so that lateral extensions can only form where there is enough material to support them).

If a termite is able to legally build at location l, it does so with a probability $P_b(C,T)$ that depends on both the cement pheromone (C) and the trail pheromone (T) concentrations in the location l. The form of the build probability



(a) From Ladley and Bullock (2004). (b) From Ladley and Bullock (2004). (c) From Ladley and Bullock (2004)

(d) 150 time steps. (e) 600 time steps. (f) 10000 time steps.

Figure 2: Replication of, and comparison with, a royal chamber construction simulation by Ladley and Bullock (2004). Termites are represented by black diamonds and cement pheromone is indicated by green shading.

is a combination of two logistic functions, each representing the influence of a pheromone type. It is given by:

$$P_b(C,T) = \left(P_{min} + \frac{P_{max} - P_{min}}{1 + e^{-k_1(C - C_{max}/2)}}\right) f(T) \quad (4)$$

where:

$$f(T) = \left(1 - \frac{1}{1 + e^{-k_2(T - T_{max}/2)}}\right) \quad (5)$$

The function, $f(T)$, allows the presence of trail pheromone to inhibit building behaviour, modulating the probability of building such that $P(C,T) \rightarrow 0$ as $T \rightarrow T_{max}$ for all C. This function is motivated by observations that termites tend not to build where trail pheromone concentration is high (Bruinsma, 1979). P_{min} and P_{max} are the minimum and maximum build probabilities in the absence of trail pheromone, and obtain when the levels of cement pheromone are 0 or C_{max}, respectively. Parameters k_1 and k_2 define the steepness of the logistic functions.

Returning After placing a pellet of building material a termite returns to the queen by following the building pheromone gradient. During this journey the termite deposits trail pheromone in each location that it moves through. The amount of trail pheromone that a termite leaves in each location decreases linearly with time so that a positive gradient which other termites can follow when finding a building location is created as simply as possible. The amount is given by:

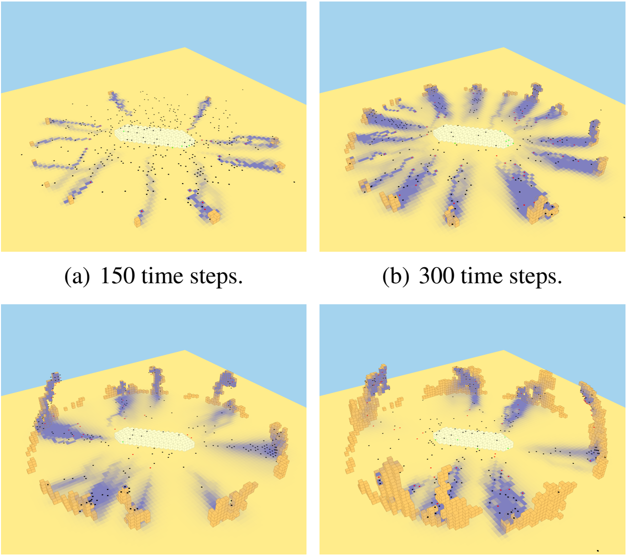
$$T(t) = \phi_T \frac{\max(L_T - t, 0)}{L_T - t}. \quad (6)$$

where ϕ_T is an initial value of trail pheromone, t is the number of time steps since the termite placed a block and L_T is a parameter defining the length of time over which each termite is able to leave a trail.

Results

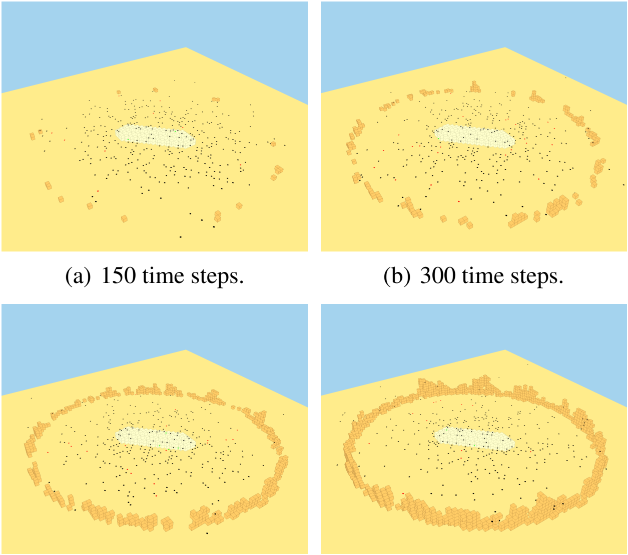
All simulations were carried out in a world of size $120 \times 120 \times 60$ locations. This is larger than the world size used by Ladley and Bullock (2004, 2005) and other parameters were adjusted accordingly so that the simulations were effectively operating at a higher resolution. The configuration of queen blocks making up the termite queen was modified so that it was twice the size of the original in all dimensions. The purpose of increasing the resolution of the simulations was to improve the separation between, and definition of, pheromone trails.

We first replicate the creation of a royal chamber in the manner of Ladley and Bullock (2004, 2005). In this earlier


(a) 150 time steps. (b) 300 time steps.

(c) 600 time steps. (d) 1200 time steps.

Figure 3: Construction of a royal chamber using building, cement and trail pheromones. Trail pheromone is indicated by purple shading.


(a) 150 time steps. (b) 300 time steps.

(c) 600 time steps. (d) 1200 time steps.

Figure 4: Attempted construction of a royal chamber when termites are prevented from leaving trail pheromone. All parameters are the same as figure 3.

model the worker termites permanently exhibit only building behaviour and follow only cement pheromone gradients to the deposition zone. Blocks of building material are placed with a fixed probability and after building each termite is removed from the world and replaced at random. Both queen and cement pheromones have the same parameters in this model, ($\phi_Q = \phi_C = 400$, $\rho_Q = \rho_C = 1/7$, $\nu_Q = \nu_C = 0.01$) and $r = 0.5$. The number of termites is $n = 400$ and the fixed build probability is $P_b = 0.1$. Q_{min} and Q_{max} are 0.1 and 0.2 respectively and the sensitivity parameter for following cement pheromone is set to $\mu_C = 0.005$. These parameters were chosen so that the replicated model output was visually a good match to the earlier model. We also follow Ladley and Bullock (2004, 2005) in allowing simulations to run for a number of time steps before introducing worker termites in order to establish a stable building pheromone template defining the deposition zone.

The new model is able to successfully replicate the output from the earlier model at higher resolution (figure 2). Initially, building is concentrated at a few sites. Mound-like pillars form at these sites, before low walls between them are constructed and eventually a dome is completed.

To explore the combined influence of building, cement and trail pheromones, and in keeping with the observations of Bruinsma (1979), the cement pheromone parameters in the new model were chosen so that the attractive influence of the pheromone from the deposition zone does not extend as far as the body of the termite queen. Trail following by the termites therefore becomes important in order to direct them to sites in the building zone that already have building material and thereby achieve a significant stigmergic effect. The value used for for w was 0.001 so that the total probability of a termite moving to a location k is much more influenced by cement pheromone if the levels of both pheromones are comparable.

The number of time steps subsequent to placing a block for which termites can leave trail pheromone, L_T, was set to 25, which is the approximate number of locations from the foot of the deposition zone to the termite queen, to ensure that the amount of trail pheromone deposited approaches 0 near the queen.

Pheromone parameters were $\phi_Q = \phi_C = \phi_T = 400$, $\rho_Q = 1/7$, $\rho_C = \rho_T = 0.017$, $\nu_Q = 0.01$, $\nu_C = 0.1$, $\nu_T = 0.01$ and $r = 0.5$. Values selected for the build probability parameters were $P_{min} = 0.1$, $P_{max} = 0.3$, $k_1 = k_2 = 0.1$, $C_{max} = \phi_C/2$ and $T_{max} = \phi_T/2$.

Example output is shown in figure 3. The pheromone trails created by the termites are clearly visible and the model retains the capacity to produce peaks and pillars that are similar to those shown in the model of Ladley and Bullock (2004, 2005). In addition, there is visible evidence of lamellae formation and at later stages of construction the termites are following the trails to the deposition zone, whereas at earlier stages their movement is more random. These features are in keeping with the observations of Bruinsma (1979).

The experiment of Bruinsma (1979) in which the termites are prevented from depositing trail pheromone is recreated in the model and the result is shown in figure 4. In this

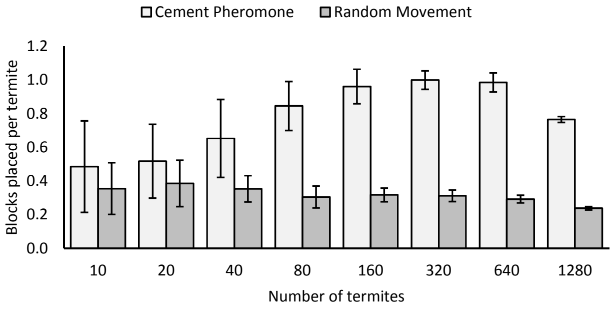


Figure 5: Recruitment curves for the replicated model of Ladley and Bullock (2004, 2005) with a world size matching the earlier model. Each value is a mean over 20 simulations, with each simulation running for 200 time steps. Error bars represent +/- one standard deviation.

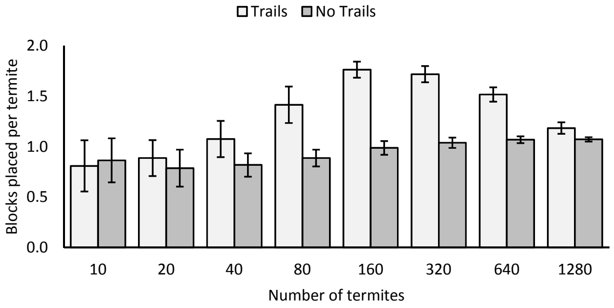


Figure 6: Recruitment curves for the new model. Each value is a mean over 20 simulations, with each simulation running for 600 time steps. Error bars represent +/- one standard deviation.

case there is visible evidence that the termite building has slowed considerably and there is no significant pillar formation, although there is still some visible peak formation at later times. The attractive range of the cement pheromone is not large enough to allow the termites to follow it from the termite queen to the deposition zone but it still has a local stigmergic effect, allowing the formation of some small vertical enhancements. This demonstrates how pillar formation might be disrupted by preventing the deposition of trail pheromone.

To demonstrate a qualitative correspondence between their model and the observations of Bruinsma (1979), Ladley and Bullock (2005) produced a graph of the mean number of blocks placed *per termite* after a fixed number of time steps had elapsed, for different termite colony sizes (i.e., different values of n). Bruinsma (1979) produced a similar graph for worker termites to show that the build rate per termite is low for small colony sizes since recruitment to building sites is not efficient at low population density, but that the stigmergic feedback effect increases this build rate as the size of the colony grows until a saturation level is reached, when there is interference between termites competing to build. Equivalent results for the replicated model are shown in figure 5.

For comparison, figure 5 also shows a graph for scenarios in which there is no cement pheromone attraction. In this case, the number of blocks placed per termite decreases slightly with increasing colony size, which we take as an indication that stigmergic activity is not present.

Figure 6 shows equivalent results for the new model. When trail pheromone deposition is allowed the graph demonstrates that a stigmergic feedback effect is present, as in the earlier model and observations. When trail pheromone deposition is prevented the blocks placed per termite appears to be more constant across colony size but still increases slightly and does not follow the same pattern as the random scenario in figure 5. This is consistent with the much reduced stigmergic effect of more localised cement pheromone attraction shown in figure 4.

Discussion

The simulations demonstrate behaviour that agrees qualitatively with observations of termite royal chamber construction and suggest a plausible mechanism through which trail pheromone may be involved in such construction.

The algorithms governing pheromone following, build probability and trail deposition are not known to be ideal and have been chosen for simplicity or plausibility. We do not know exactly how or when the trails of termites building a royal chamber are laid, if they need to be reinforced in both directions or if perhaps the strength of the pheromone trail laid is dependent on the amount of cement pheromone detected before building, for example. Implementing the model on a three-dimensional cubic lattice is also not ideal because the anisotropic nature of the neighbour distribution makes the design of a trail following algorithm difficult. The introduction of the restriction preventing termites from walking back over their last few previous locations was an attempt to alleviate this difficulty.

Furthermore, there are several additional factors involved in termite construction that have not been included in the model but which may influence building behaviour, such as gravity, temperature or humidity. Termites are also known to be influenced by tactile stimuli, so that sharp angles and existing obstacles incite building activity (Stuart, 1967; Bruinsma, 1979).

One explanation for why the model is able to show narrower pillar formation and lateral growth and the earlier model of Ladley and Bullock (2004, 2005) was not is the fact that, as well as directing termites to the deposition zone, trail pheromone also supresses building activity. This means that after some pellets of building material have been placed at lower elevations there is enough trail pheromone at these elevations to reduce the probability of further building. The termites therefore effectively have to move upwards to find suitable build locations, and once at higher elevations they are not prevented from building laterally.

However, once enough termites are building at higher elevations, the concentration of trail pheromone at lower elevations becomes insufficient to prevent building and additional construction occurs from the ground up once more. This effect is exacerbated by the termites effectively getting stuck for a number of time steps at higher elevations as they have no concept of 'down'. During this time the trails weaken and subsequent building becomes more spread out.

The next step is to investigate objective, quantitative measures of the model outcomes and processes in order that it can be automatically determined whether or not a model run is successful. These quantitative measures will be necessary in order to use a sensitivity analysis or statistical emulators to determine the influence of model parameters. Methods for quantifying the model output could involve the use of autocorrelation functions, entropy measures, mathematical morphology (Serra, 1982) or measures of clustering, for example. Given the shape of the royal chamber and the binary nature of the block distribution in three dimensions (locations either have a block or do not), the application of such methods presents a challenge.

Some simpler initial analysis of the effects of the model parameters on factors such as the change in the distribution of build probabilities with time step or the minimum rate of block placement required to maintain a desired attractive radius of cement pheromone may help to identify key parameter ranges.

Conclusion

A new three-dimensional agent-based model has been established that is able to reproduce earlier work by Ladley and Bullock (2004, 2005). Moreover, by extending the model to include a behavioural cycle documented by Bruinsma (1979), we have been able to qualitatively demonstrate that pillar formation can also occur when the movement and building behaviours of termites are influenced by a combination of both cement and trail pheromones. This three-pheromone model additionally shows lateral growth from pillars (lamellae), movement from the termite queen to the deposition zone that becomes more direct as trails form, and a distribution of building material that has significantly suppressed pillar formation when trail pheromone deposition is prevented. These features are consistent with termite royal chamber construction (Bruinsma, 1979) and have not been simulated in earlier models (Deneubourg, 1977; Courtois and Heymans, 1991; Bonabeau et al., 1998; Ladley and Bullock, 2004, 2005).

The next step is the quantitative analysis of the model parameters, algorithms and output in order to identify redundancies in the parameter space, and regimes or critical points in the dynamics of the model and to distinguish between more or less successful simulations of royal chamber construction.

Acknowledgements

This work was supported by an EPSRC Doctoral Training Centre grant (EP/G03690X/1).

References

Bonabeau, E., Theraulaz, G., Deneubourg, J.-L., Franks, N. R., Rafelsberger, O., Joly, J.-L., and Blanco, S. (1998). A model for the emergence of pillars, walls and royal chambers in termite nests. *Philosophical Transactions of the Royal Society of London B*, 353(1375):1561–1576.

Bruinsma, O. H. (1979). *An analysis of building behaviour of the termite macrotermes subhyalinus (rambur)*. PhD thesis, Wageningen University.

Courtois, P.-J. and Heymans, F. (1991). A simulation of the construction process of a termite nest. *Journal of Theoretical Biology*, 153(4):469–475.

Deneubourg, J.-L. (1977). Application de l'ordre par fluctuations a la description de certaines étapes de la construction du nid chez les termites. *Insectes Sociaux*, 24(2):117–130.

Doursat, R., Sayama, H., and Michel, O., editors (2012). *Morphogenetic Engineering: Toward Programmable Complex Systems*. Springer.

Grassé, P.-P. (1959). La reconstruction du nid et les coordinations inter-individuelles chez bellicositermes natalensis et cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, 6(1):41–80.

Hirsch, C. (1988). *Numerical Computation of Internal and External Flows Volume 1: Fundamentals of Numerical Discretization*. Wiley-Interscience.

Howse, P. E. (1970). *Termites: A Study in Social Behaviour*. Hutchinson & Co Ltd.

Ladley, D. and Bullock, S. (2004). Logistic constraints on 3d termite construction. In *Ant Colony Optimization and Swarm Intelligence (Proceedings of ANTS 2004)*, pages 178–189. Springer.

Ladley, D. and Bullock, S. (2005). The role of logistic constraints in termite construction of chambers and tunnels. *Journal of Theoretical Biology*, 234(4):551–564.

Leuthold, R. H., Bruinsma, O. H., and van Huis, A. (1976). Optical and pheromonal orientation and memory for homing distance in the harvester termite *hodotermes mossambicus* (hagen). *Behavioral Ecology and Sociobiology*, 1(2):127–139.

Serra, J., editor (1982). *Image Analysis and Mathematical Morphology*. Academic Press, Inc.

Stuart, A. M. (1967). Alarm, defense, and construction behavior relationships in termites (isoptera). *Science*, 156(3778):1123–1125.

Stuart, A. M. (1970). The role of chemicals in termite communication. In *Advances in chemoreception volume 1: Communication by chemical signals*, pages 79–106. Appleton-Century-Crofts.

Wilson, E. O. (1971). *The Insect Societies*. Harvard University Press.

From tadpole to frog: artificial metamorphosis as a method of evolving self-reconfiguring robots

Michał Joachimczak, Reiji Suzuki, Takaya Arita

Graduate School of Information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
mjoach@alife.cs.is.nagoya-u.ac.jp

Abstract

We show how the concept of metamorphosis, together with a biologically inspired model of multicellular development can be used to evolve soft-bodied robots that are highly adapted to two radically different environments (e.g., aquatic and terrestrial). Each evolved solution defines two pairs of morphologies and controllers, together with a process of transforming one pair into the other. Animats develop from a single cell and through divisions and deaths reach their initial "larval" form adapted to the first environment. To obtain "adult" form adapted to the second environment, the larva undergoes metamorphosis during which new cells are added or removed and its controller is modified. Importantly, our approach assumes nothing about what morphologies or methods of locomotion are preferred. Instead, it successfully searches the vast space of possible designs and comes up with complex, life-like solutions *de novo*.

In this paper, we describe the approach we employ and present examples of metamorphic soft-robots. We compare two different approaches to evolving aquatic and terrestrial animats, investigate evolved motion strategies, the process of metamorphosis and its evolution.

Introduction

Metamorphosis is a process during which an organism that has already finished its embryonic development undergoes a relatively fast and considerable change to its body structure through cell growth, differentiation and death. It is often accompanied (or actually necessitated) by a change of environment the organism lives in. Metamorphosis is exhibited by a wide variety of taxa as different as insects, mollusks or amphibians. In the latter case it testifies to amphibians' evolutionary ancestry: land based amphibians begin their lives in aquatic environment and develop first into a fish-like larval stage (a tadpole). Maturing tadpoles undergo metamorphosis which allows them to switch to a terrestrial habitat. This involves changes such as the loss (reabsorption) of gills, tail, lateral-line system and the gradual growth of jaw and limbs. Importantly, the processes that occur during metamorphosis are the very same processes that shape organism during its embryonic growth. Metamorphosis is a manifestation of multicellular development and just like all development, it is deeply tied to species evolutionary history (see, e.g., Carroll et al., 2004).

The field of artificial embryogeny, to which this work belongs, attempts to capture the seemingly endless capability of nature to generate forms by reproducing key properties of development *in silico*. This typically involves bio-inspired construction process in which a structure (such as a robot's body) is progressively built from smaller elements. Depending on the chosen level of abstraction this may involve elements such as rods (Komosinski and Rotaru-Varga, 2002), primitives and joints (Sims, 1994; Pilat et al., 2012), or artificial cells, as in the case of the system employed by us and related ones (Dellaert and Beer, 1996; Eggenberger Hotz, 1997; Schramm and Sendhoff, 2011; Bongard and Pfeifer, 2003). Self-assembly from higher level components such as blocks and joints or using an even higher level abstraction of development such as CPPN (e.g., Cheney et al., 2013) has been demonstrated to be an effective way to generate interesting robotic designs. In our line of work, however, we aim to explore the potential and scalability of a much more biologically inclined and more fine-grained artificial development, where arbitrary morphologies can be freely assembled from large numbers of cells (hundreds, thousands), each taking independent decisions about their fate and interacting through simulated physics of developmental environment. So far we have demonstrated how this approach allows us to evolve a rich variety of complex soft-bodied animats with emergent higher level features such as appendages constructed of dozens of cells and acting as artificial legs (Joachimczak et al., 2014).

In this paper we show how the concept of metamorphosis can enhance artificial development by allowing evolutionary algorithm to automatically produce solutions (here, soft-robots) that can take two potentially very different forms, each adapted to its target environment. Importantly, one form can transform into another, offering exciting potential for designing robots that could efficiently operate in radically different environments: a rescue robot launched for a mission from sea could swim to a shore, transform into terrestrial form and continue its mission on its newly grown legs.

The idea of combining metamorphosis with an evolutionary algorithm is by itself not new. For example, Bongard (2011) demonstrated how morphological change during a legged robot's lifetime (progressive extension of its legs) facilitates evolution of higher quality gait controllers. The nature of the change and the morphology were however predefined. In another, more conceptually similar work (Tufte,

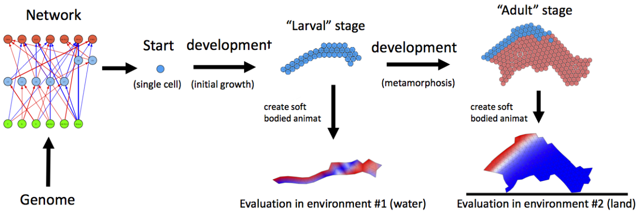

Network
Start
development
(single cell)
(initial growth)
"Larval" stage
development
(metamorphosis)
"Adult" stage
create soft bodied animat
create soft bodied animat
Genome
Evaluation in environment #1 (water)
Evaluation in environment #2 (land)

Figure 1: Conceptual overview of single genome's evaluation allowing for evolution of metamorphic individuals.

2011) metamorphosis was used to evolve simple digital circuits implemented as cellular automata and producing new functionality through rearrangement. To our knowledge, this is however the first case of using metamorphosis and evolution to automatically discover robots that have forms adapted to two different environments, without specifying desired morphologies or the nature of the transformation.

Developmental model

In the following sections we provide a short overview of the essential concepts behind our artificial embryogeny approach to evolving soft-bodied robots. It is a refined version of the system we introduced in Joachimczak et al. (2014), of which some aspects were simplified in order to further reduce complexity (both conceptual and computational). Namely, we allowed for only non-recurrent networks that control cells' behaviour, only one type of gene activation function (sigmoidal) and we also simplified mechanism of actuation (and made it similar with the one in Joachimczak and Wróbel, 2012). We believe an even simpler and more straightforward design of our developmental system makes for a more convincing argument about applicability of metamorphosis approach to other developmental systems.

We assume that each animat begins its life as a single cell and grows through subsequent cellular divisions and deaths (apoptosis). The fate of every cell is determined by a shared control mechanism: a simple abstraction of gene regulatory network (GRN) in the form of a feed forward neural network. Despite being controlled by the same network, cells will act differently as the external signals that are fed to the inputs of the network depend on cell's position in the growing embryo, as well as signals received from neighbouring cells. After the development has finished, the resultant multicellular structure is used as a template for a morphology of soft-bodied animat that can move by contracting and expanding regions of the body surrounding each cell (Fig. 1).

Physics of development Development takes place in a continuous 2-D space. Cells are represented as discs and undergo elastic collisions simulated with springs. A cell's physical state is defined by its position, its velocity, and orientation vector which determines the direction of division. All cells have uniform size and mass. Springs connect only the nearest neighbours and are determined dynamically, as the embryo grows, with the resting length set to the sum of attached cells' radii. We use Delaunay triangulation to determine the connectivity between cells and then remove links longer than 150% of cell's diameter. Disjoint structures are prevented from occurring (see Joachimczak et al., 2014, for details).

Control gene network The control network present in all cells has 6 inputs in total. This includes a fixed bias signal and time signal that is linearly increasing from 0 to 1 over the course of development. Given that the networks governing cellular behaviour in the presented experiments are stateless, spatially differentiated signals are the only way for cells to take on different behaviours. We have provided a simple, maternal gradient-like mechanism in the form of cell's X and Y coordinates available as inputs to the network. Additionally, to simulate simple morphogens produced by cells themselves, the control network has two "morphogen" outputs and two associated "morphogen" inputs. For any given cell, the activation of the latter is set to be an average of the corresponding morphogen outputs of its neighbours. Apart from the two above mentioned outputs, cells have 5 other outputs that determine actions taken by them. This includes an output whose non zero activity prevents division, a signal that controls the angle of division, a signal that causes aptoptosis (cell's death) and two outputs that determine frequency and phase shift of "muscle" contractions during the locomotion stage.

The control networks were updated in all cells synchronously. This involved setting the state of input nodes and propagating signals the number of steps equal to the longest path between input and output found in the network.

Cell division and death All cells are bound to divide with each subsequent update of gene regulatory network (occurring every 30 steps of developmental simulation) unless the output inhibiting division is active. Division is allowed to occur if and only if the space in the direction of division is not occupied already by another cell (the rationale for this approach can be found in Joachimczak et al., 2014).

The newly created cell is placed next to the original cell in the direction determined by its orientation vector. Its angle is determined at the moment of division, based on the state of the associated output of regulatory network. The angle is interpreted as relative to the angles of cell's nearest neighbours and thus, unless the state is different from zero, all cells will simply divide in the same direction. The new cell is controlled by the same network, hence any change in the patterns of network activities is the result of symmetry being broken owing to the differences in external signals perceived by each of the cells.

Apoptosis (cellular death) occurs whenever the state of associated network output is found to be above zero and leads to the cell being removed from the embryo.

We used a hard limit of 256 cells and, additionally, we penalized individuals that have created more than 1024 cells during their development by reducing their fitness value by 90%. We did so to limit the occurrence of "wasteful" solutions in which cell creation and death is kept in balance and never stops. While we have shown in our previous work (Joachimczak et al., 2014) that solutions that self-terminate growth are not difficult to evolve in our approach, they come at a cost, both computational and in reduced quality of obtained solutions. Although promoting self-termination facilitates emergence of beneficial properties of development such as increased robustness (Devert et al., 2011), in this work we opted for the simple and more manageable scenario, in which evolution can exploit the hard limits of developmental time and cell count to stop development.

Soft-bodied locomotion

The multicellular morphology of an embryo obtained at the end of developmental stage was used as a template for a soft-bodied animat that was evaluated for its capability to move in a simulated environment during locomotion stage. Although both representations are a spring-mass system and rely on the same physics simulation engine, the physical constants and the rules that govern reshaping of each structure differ between the stages. Namely, development occurs always without gravity in a fluid-like environment and involves continuous reorganization of connectivity between cells. On the other hand, during the locomotion stage, cells no longer divide or die, and the connectivity between the cells remains fixed (the animat however does undergo elastic changes).

More precisely, during the locomotion stage animat is represented as a spring-mass system with point masses located at the centres of cells in the embryo and with springs forming a triangular mesh. The outline of the shape is defined by the outer cells of the embryo, while the inner part of the animat is triangulated using Delaunay algorithm, as a way of approximating uniformly elastic material. As this approach can produce a morphology with a protrusion that is connected by a single spring to the main body, such protrusions are cut off from the soft-body representation.

The resting lengths of springs are assigned based on the distances between cell centres at the end of developmental stage. Springs in our system are governed by the Hooke's law with damping. Additionally, each triangular region has an equilibrium pressure S_0 (determined based on its surface area at the end of development) providing animat with a hydrostatic skeleton and preventing excessive compression or stretching of body regions. All springs share the same Hooke's constant value k but can have different resting lengths. To avoid self penetration of animat bodies, masses representing cells undergo elastic collisions with springs.

Actuation is achieved by modifying resting lengths of springs attached to a given cell. This results in the region around the cell contracting or expanding. The change in length is driven by a sinusoidal oscillation pattern associated with every cell. The period of oscillation and the phase shift for every cell is determined by 2 corresponding GRN outputs at the end of developmental stage. During locomotion, the default resting length L_0 of a given spring is modified according to:

$$L = (1 + A\sin(\frac{2\pi t}{T_1} + \phi_1) + A\sin(\frac{2\pi t}{T_2} + \phi_2)) \cdot L_0 \quad (1)$$

where t is simulation time, A is the amplitude ($A = 0.2$), T_1, T_2 are the evolved periods of oscillation (scaled to span desired range), and ϕ_1, ϕ_2 are evolved phase shifts (scaled to $-\pi$ to π). To prevent sudden changes in resting length for cells with non-zero phase shift at the start of the locomotion stage, the amplitude is progressively increased during a short "warm-up" period.

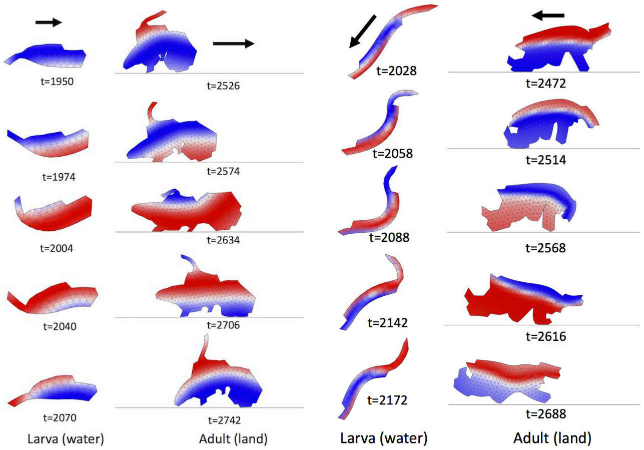
Terrestrial environment was constructed by placing animats on top of a horizontal flat surface and introducing gravity and friction between animat's nodes and the surface. For a fluid based environment gravity was disabled and fluid drag was introduced. We used fluid drag model based on Sfakiotakis and Tsakiris (2006) which assumes that the fluid is stationary and that the force acting on a single edge on the outline of the body is the sum of tangential and normal drag components for the motion of this edge against fluid.

Genetic encoding and genetic algorithm

The neural network model and genetic representation is based on the MultiNEAT library (Chervenski and Ryan, 2014, see reference for the configuration file used), the implementation of the NEAT evolutionary algorithm (Stanley and Miikkulainen, 2002). In the NEAT method networks are represented in the genomes as a list of nodes and their types (input, output, normal) and a list of connections. The algorithm keeps track of innovations history and uses it to perform crossover. It also uses fitness sharing approach with the goal of preserving diversity and protecting new solutions before they have to compete with the rest of the population. We used population sizes of 300 and runs of 2000 generations. Initial population was created as a fully connected feed-forward network with a hidden layer and random weights. To prevent emergence of recurrent networks, mutations that created cycles in the network were rejected.

Evolution of metamorphic animats

We base our approach to evolving metamorphic individuals on a simple idea: we evolve growing multicellular animats similarly to how we would evolve morphologies and controllers capable of locomotion in a given environment in our previous work. However, rather than evaluate animat's morphology performance in a virtual environment after its development is finished, we evaluate it twice, at different stages of development. More precisely, we allow each genome to control embryonic development for 600 time steps. The performance of emerged morphology is evaluated in the first environment (e.g., for its capability to swim). Next, the de-



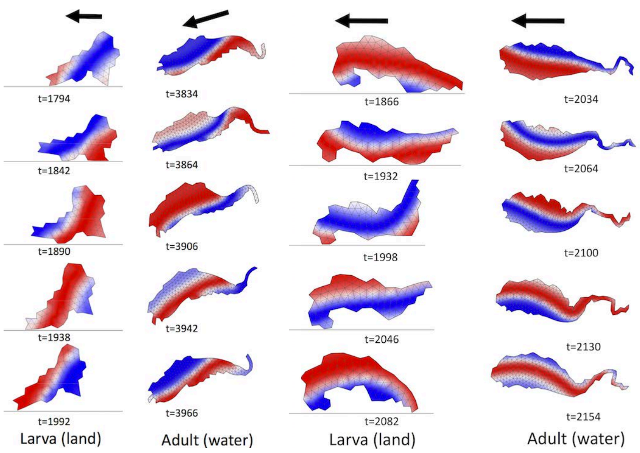
(a) (b)

Figure 2: Motion snapshots of two solutions (a,b) evolved to move first in aquatic environment and then in terrestrial environment (left: before metamorphosis, right: after metamorphosis). Arrows indicate direction of movement, snapshots were selected to represent one motion cycle. Larval and adult stage are not to scale (larva is smaller). Colours represent whether the region of the body is currently expanded (red), contracted (blue) or at its resting size (white). Videos for (a) can be accessed at http://goo.gl/GozK89 and http://goo.gl/19rJAf, videos for (b) at http://goo.gl/aQUbPj and http://goo.gl/nM47Lb

velopment is allowed to continue for another 600 time steps and the resulting morphology is evaluated one more time in the second environment. Importantly, beyond defining the multicellular growth and locomotion models, our approach assumes nothing (other than the limits of the system) about desired size or complexity of the animats, nor the type of locomotion that is preferred for each environment. It also does not explicitly assume that metamorphosis has to happen if development is continued beyond the "larval" stage.

The discontinuity between developmental and locomotion stages that is inherent to our approach necessitates some trade-offs. After the "larval" stage has been evaluated for its performance as a soft-bodied robot, the system needs to return to the developmental stage to continue growth. If development was simply allowed to continue after the positions of cells have been rearranged during locomotion, the course of growth would be highly influenced by the temporal configuration of cells at the end of evaluation in the first environment. As we considered this undesirable and likely deceptive for evolutionary search, we chose to simply resume development from the exact state it was before entering the first locomotion stage, disregarding any elastic changes that occur during locomotion. Fig. 1 provides a concise summary of our approach.

Given the need to optimize individuals with respect to two objectives (the performance of larva and adult) the fitness was determined through a simple scalarization: $f = \sqrt{d_L d_A}$, where d_L, d_A were distances achieved by larval and adult stages, respectively. While we expect an algorithm dedicated to solving multi-objective problems would be advantageous, we opted for the simple scalarization as it allowed us to perform the experiments without the need to modify the NEAT library we were using (Chervenski and Ryan, 2014).



(a) (b)

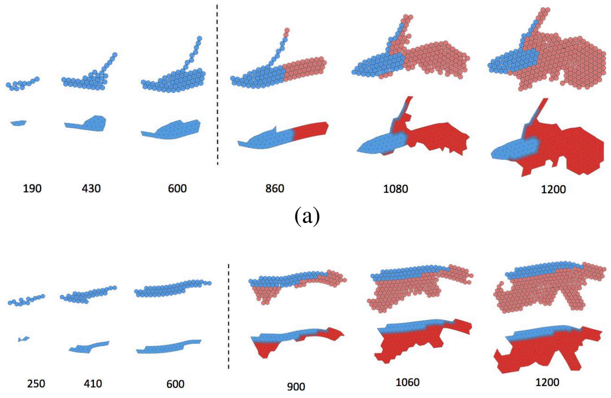
Figure 3: Motion snapshots of two solutions (a,b) evolved to move first in terrestrial environment and then in aquatic environment (left: before metamorphosis, right: after metamorphosis). Arrows indicate direction of movement, snapshots were selected to represent one motion cycle. Larval and adult stage are not to scale (larva is smaller). Colours represent whether the region of the body is currently expanded (red), contracted (blue) or at its resting size (white). Videos for (a) can be accessed at http://goo.gl/XZLVtx and http://goo.gl/DgLI97, videos for (b) at http://goo.gl/VTZYUV and http://goo.gl/o2r5bh

Results

We performed two types of experiments. The aim in both cases was to evolve individuals that would be adapted to swimming in a fluid environment and moving in a land based one at different stages of their development. In the first scenario, we evaluated developed individuals first in an aquatic environment (as a larva), allowed them to continue growth (hence, potentially undergo metamorphosis) and then evaluated them again in a terrestrial one (as an adult). In the second scenario the order of stages was reversed: we evaluated embryos first in terrestrial environment and then in the aquatic one. Each type of experiment consisted of 20 independent evolutionary runs (using different random generator seeds).

Evolved morphologies

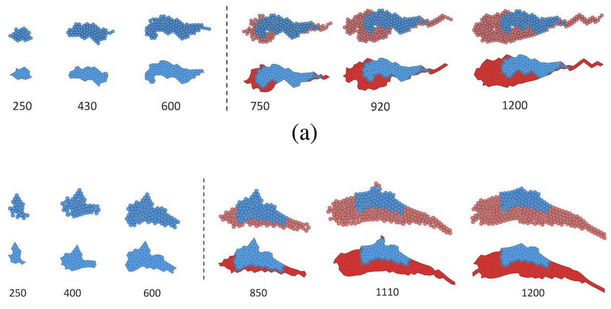
All of the evolutionary runs we performed resulted in individuals undergoing metamorphosis, i.e., they would employ different morphologies for each of the two types of environments they were tested in. The fact that growth occurred



(a)

(b)

Figure 4: Developmental processes of the two individuals shown in Fig. 2. Dashed line separates larval stage from the beginning of metamorphosis. Upper rows show multicellular representation that is used during development, bottom rows show a preview of soft-bodied representation (note that protrusions with a thickness of 1 cell only are removed during this transformation). Soft bodied representation is however used only at the end of each developmental stage, i.e., at 600 and 1200 time steps. Labels indicate developmental time. Blue colour depicts cells that were created during the larval stage, red depicts cells created during metamorphosis. Videos can be accessed at: http://goo.gl/ejHz9T and http://goo.gl/929Ch6



(a)

(b)

Figure 5: Developmental processes of the two individuals from reversed stages experiment shown in Fig. 3. Dashed line separates larval stage from the beginning of metamorphosis. Upper rows show multicellular representation that is used during development, bottom rows show a preview of soft-bodied representation. Soft bodied representation is however used only at the end of each developmental stage, i.e., at 600 and 1200 time steps. Labels indicate developmental time. Blue colour depicts cells that were created during the larval stage, red depicts cells created during metamorphosis. Videos can be accessed at: http://goo.gl/3pqksv and http://goo.gl/YnwCFu

when the developmental process was allowed to continue is not surprising by itself as, unless the growth inhibition genes were active, the cells would continue to divide. Since, contrary to our earlier work (Joachimczak et al., 2014), we did not apply evolutionary pressure to self-terminate growth, continued divisions were the most likely outcome.

What was however surprising was to observe how individuals adapted their "larval" and "adult" stages to each environment. Two examples of such adaptation (belonging to the most successful individuals obtained in 20 evolutionary runs) are shown in Fig. 2 (see also Fig. 4 for their development, discussed in the next section). Their "larval" stages, having elongated, streamlined morphologies are clearly adapted to swimming in water environments and move with an undulatory, fish-like motion pattern. The pattern itself is a result of waves of cellular contractions that travel through the body in the direction perpendicular to that of movement. On the other hand, their "adult" stages clearly display non-trivial adaptation to land locomotion, having grown leg-like appendages to support their body and moving by cyclically shifting their centre of weight between them. We find it remarkable that an artificial evolution system converges to solutions reminiscent of amphibian lifecycles, with a tadpole-like stage that undergoes metamorphosis by shedding part of its tail and growing appendages.

To find out whether the observed morphologies were a result of evolutionary pressure or a simple side effect of some property of our developmental model (e.g., the bias towards continued growth), we performed the same experiment with environments for each stage inverted: larva had to move on land and adult had to be able to move in the water environment. Fig. 3 shows results of two evolutionary runs that led to high fitness individuals. We were able to observe how, again, locomotion relying on simple appendages emerged in the land based environment, whereas the aquatic adult forms displayed elongated morphologies and undulatory patterns of locomotion. This result indicates high evolvability of our approach, as it has no problem discovering legged locomotion and undulatory swimming regardless of the order of developmental stages.

Finally, although the direction of motion between larval and adult stage is similar in the figures shown, it was not uniformly the case in other evolutionary runs. However, since our virtual animats do not have any sensors and therefore no concept of front or back, there is no direct reason for evolution to maintain the direction of movement unchanged.

Development and metamorphosis

In order to gain insight into how animats undergo metamorphosis we visualized developmental processes of the 4 individuals depicted in Figs. 2 and 3. In each case (Figs. 4 and 5), metamorphosis proceeds by adding cells to the larval stage, resulting in a structure that is considerably larger and has the previous developmental stage (or a part of it) "embedded" inside. Apoptosis occurring during metamorphosis was not uncommon and can be observed in the individual seen in Fig. 4a (note the loss of the "tail" on the left side by comparing shapes at $t = 860$ and $t = 1080$) and in the individual in Fig. 5b (removal of cells at the top of the body, compare shapes at $t = 850$ and $t = 1200$).

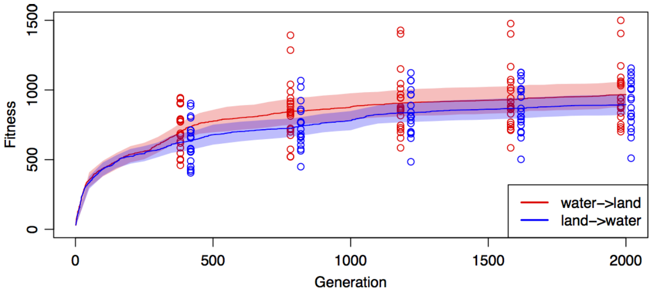


Figure 6: Fitness of the best individual in population over evolutionary time for the two types of experiments in evolving metamorphic individuals. Lines show average fitnesses of 20 independent evolutionary runs each, colour stripes represent 95%, bootstrapped confidence intervals for the averages.

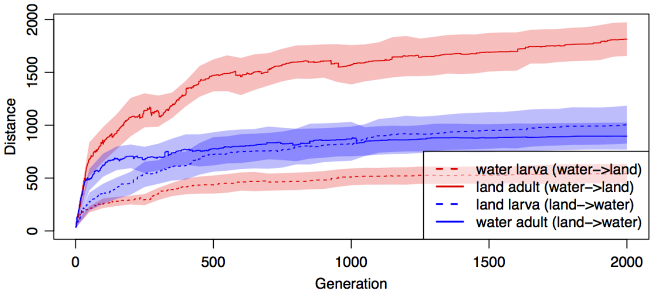


Figure 7: Distances achieved by each developmental stage of the best individual in population for both types of experiments. Lines show average distances from 20 independent evolutionary runs of each experiment type, colour stripes represent 95%, bootstrapped confidence intervals for the averages. Dashed line represents "larval" stage performance, solid represents "adult" stage.

While the metamorphosis of a water larva to a land based adult involved growth of appendages, the metamorphosis of land larva to an aquatic adult involved growth process that filled the space between the appendages, producing a more streamlined shape while at the same time growing tail.

The adult stage being larger than the larval form was universally observed and, as suggested previously, can be largely explained by the developmental model being biased towards non self-terminating growth.

Analysis of evolutionary runs

In order to obtain a more quantitative overview of the evolutionary trends occurring in our experiments, we analysed multiple independent evolutionary runs. Fig. 6 shows how fitness changed over evolutionary time for both types of experiments. Given the fitness function being a square root of the product of distances achieved by each developmental stage in their respective environments, the fitnesses obtained in both types of experiments are of the same level, with some potential for improvement given longer evolutionary runs and, possibly, a slight fitness advantage in the second type of experiment.

However, analysing performance of each developmental phase independently paints a very different picture (Fig. 7). The "from water to land" scenario achieves its fitness owing to swimmers that swim small distance and very efficient runners, capable of traversing on average 4 times larger distance. On the other hand, the "from land to water" scenario achieves its fitness owing to much better swimmers and much weaker runners, traversing about the same distance in their respective environments. What is the source of this discrepancy? We suspect it can be explained by a combination of properties of simulated physics and biases introduced by the developmental model. For one, given that there are no costs involved in growing/actuating larger bodies, being bigger allows to traverse longer distances. More precisely, in the water, the available power of actuators will be proportional to the number of cells (roughly equivalent to the 2-D animat's surface area) whereas the fluid drag will be proportional to the body's cross section (roughly proportional to the square root of the surface area). Similarly, in terrestrial environment, the amount of friction that needs to be overcome will on average be proportional to the perimeter length. Thus, as long as fitness is calculated as absolute distance (rather than, e.g., in relation to body size), the forces exerted by animats increase faster with body size than drag forces do. Furthermore, our experiments with evolving specialized individuals (i.e., adapted to only a single environment) suggest that individuals with the same size limit achieve, on average, higher absolute distances on land than in the virtual fluid. Therefore, with the bias for "adult" stage being bigger than "larva" and ground based locomotion allowing to reach higher speeds in general, the experiments in which "adult" form is terrestrial further amplify the speed advantages of ground locomotion, whereas the experiments where the "larva" moves on land (and thus is smaller), reduce the speed advantage of ground based locomotion. Finally, it is possible that this result indicates that the evolutionary acquirement of developmental path from land larvae to a water adult is generally more difficult, however, given the biases present in the system, without additional experiments, the biases are the parsimonious explanation of the observed results.

Finally, to better understand the scope of change that happens during metamorphosis, we analysed how many cells are created and removed during the transformation. We plotted the result over evolutionary time in Figs. 9 and 8. The number of cells created varied greatly between the runs, reaching values between 90 and almost 400 (note that 256 cells were the maximum allowed, therefore a number that high also implies a large number of cells removed). First 200 generations would show a steady growth in the amount of cells added during metamorphosis and likely indicate period in which first functional metamorphic individuals are being discovered. Since all cells grown during metamorphosis originate from the cells in the larva, scaling the number of cells by its size would reveal a flatter curve (not shown). Be-

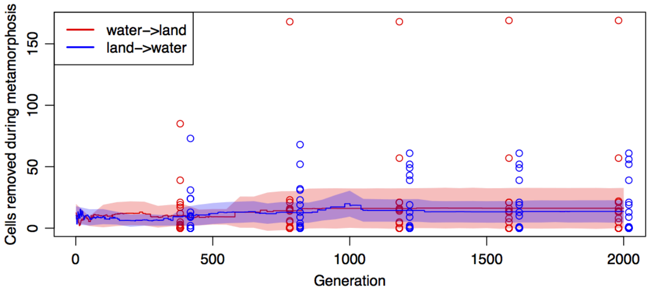


Figure 8: The number of cells that undergo apoptosis during metamorphosis over evolutionary time for the two types of experiments in evolving metamorphic individuals. Lines show averages of 20 independent evolutionary runs each, colour stripes represent 95%, bootstrapped confidence intervals for the averages.

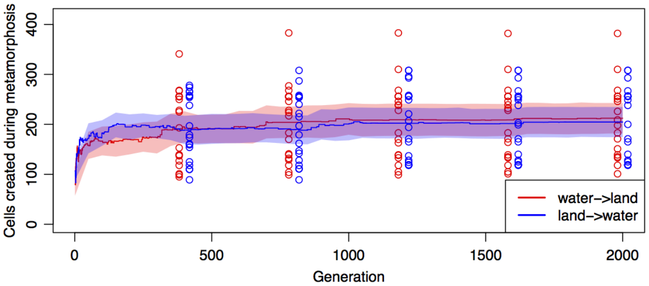


Figure 9: The number of cells that are added during metamorphosis over evolutionary time for the two types of experiments in evolving metamorphic individuals. Lines show averages of 20 independent evolutionary runs each, colour stripes represent 95%, bootstrapped confidence intervals for the averages.

yond first 500 generations there is little difference between the two types experiments in absolute terms, with both displaying an average of around 200 cells added during metamorphosis and a very large discrepancy between the runs. This however meant adults being on average around 5 times larger than larva in the "water to land" scenario and 3.5 times larger in the reversed one.

The amount of cells that are removed during metamorphosis also displays very high variation between the evolutionary runs with values between 0 and 75 (more than 150 in one case). Although not occurring in all evolutionary runs, apoptosis was a common phenomenon during metamorphosis in both types of experiments. This is a highly interesting result, as it indicates that evolution is eager to adapts apoptosis as a method of reshaping morphology in our system, a process well known to be essential in the growth of biological organisms.

Summary and future work

In this preliminary work, we have shown how it is possible to evolve soft-robots that are adapted to two different environments by being able to transform their morphology from one form to another. Doing so required relatively small changes on top of artificial development system we have recently introduced (Joachimczak et al., 2014). Evolving metamorphic individuals turned out to be a relatively easy task for the evolutionary algorithm and, remarkably, the scope of changes that occurred during metamorphosis from aquatic "larvas" to terrestrial "adults" commonly included growth of supporting leg-like appendages, as well as a loss of "tail". Changes that, at least at a superficial level, are surprisingly similar to changes that occur during life cycles of amphibians such as frogs and toads. On the other hand, attempting to evolve individuals in which "larva" is terrestrial and "adult" form is aquatic resulted in solutions that once again discovered appendages as a method of locomotion on land, to later grow cells around appendages and produce streamlined, elongated shapes well adapted to swimming. The fact that evolved types of morphologies and motion patterns were roughly common between two cases of experiments suggests, on one hand, that these morphologies are a result of evolution converging to effective modes of locomotion rather than just an idiosyncrasy of the developmental model we employ. On the other, the way to obtain high fitness in each case was very different (Fig. 7). This means that the environmental dynamics in the life history can significantly affect the evolution of metamorphic and survival strategy due to the existence of developmental constraints, even if a highly efficient type of morphology for each stage of life exists.

The methodology presented in this paper can be seen as a way of evolving dual designs in which one genotype encodes two different specialized phenotypes, together with a method of reconfiguring one morphology into another. Coupled with an adequate modular substrate, such as the recently presented swarms of inexpensive small robots (Rubenstein et al., 2014), our approach offers exciting potential of automatically discovering morphologies consisting of hundreds of smaller, uniform cell-like components together with methods of self-assembling and transforming them to reflect changes occurring in their environments.

At the same time, given that the essential factors of metamorphic processes in multicellular organisms are incorporated into our abstract model, we believe it can also contribute to the better understanding of origin and evolution of metamorphosis. Furthermore, as it would require only a simple change of evolutionary pressures to model a progressive transformation of aquatic population into a land dwelling one, our approach could be used as a basis for a further, artificial life study that would help us understand the relationship between genetic and morphological changes that occur during major evolutionary transitions, such as when our fish ancestors started to adapt to living on land.

One aspect of artificial metamorphosis that we plan to further explore relates to the fact that genetic machinery responsible for the two morphologies of a metamorphic individual is shared. As such, it must introduce certain tradeoffs and limitations on the designs that could be achieved using this approach. After all, the morphology grown for the first stage of life has to not only be adapted to its specific environment, but also be "malleable" to environment it has to perform in during the second stage of life. Comparing our results with performance of individuals that were evolved to be specialized for each type of environment, or were evolved to be robust in both, will shed light on the cost of this malleability. Furthermore, we can envision scenarios in which the necessity to evolve "malleable" designs would actually make it easier to evolve useful solutions by directing evolution towards promising regions in the search space. For example, if the goal was to evolve individuals that can run on a horizontal surface and uphill, the morphology of good solutions for each case may be very similar and differ just by the length of appendages. Metamorphic solution could then potentially be found more easily than two separate solutions. This is because in the case of attempting to evolve a specialized individual for running uphill, the evolutionary algorithm is likely to have a problem bootstrapping the search (by finding initial individuals that move up rather than simply roll down the slope). It may also be the case that metamorphic individuals, even if not showing better performance, may be better in some respects, e.g., be more robust or having higher quality gaits, as shown in Bongard (2011). We plan to investigate this in our future work, together with investigating other methods of evolving metamorphic individuals, such as novelty search (Lehman and Stanley, 2011).

The metamorphic robots introduced in this paper undergo a single transformation. It would however be straightforward to extend this approach to a larger number of stages (or consider ability to return to a previous one). Nonetheless, having a single transformation would easily find its applications, for example in sea deployed rescue robots bound for a mission on land.

Finally, we would like to note that the potential of using artificial metamorphosis is tightly coupled with the use of artificial development and, essentially, its natural consequence. As such, artificial metamorphosis should be applicable to many other artificial embryogeny systems, further revealing the hidden potential of this highly biologically inspired approach to automatic design.

Acknowledgements

This work was supported by the Japan Society for the Promotion of Science through the JSPS Fellowship for Foreign Researchers and JSPS KAKENHI Grant Number 26-04349. High performance computing resources were provided by the Interdisciplinary Center for Molecular and Mathematical Modeling (University of Warsaw) and the Tri-city Academic Computer Center (TASK). CGAL library was used.

References

Bongard, J. (2011). Morphological change in machines accelerates the evolution of robust behavior. *Proc. Natl. Acad. Sci. U. S. A.*, 108(4):1234–1239.

Bongard, J. C. and Pfeifer, R. (2003). Evolving complete agents using artificial ontogeny. In *Morpho-functional Machines: The New Species*, pages 237–258. Springer Japan.

Carroll, S., Grenier, J., and Weatherbee, S. (2004). *From DNA to Diversity: Molecular Genetics and the Evolution of Animal Design*. Wiley-Blackwell, second edition.

Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In *Proc. of the Genetic and Evolutionary Computation (GECCO)*, pages 167–174. ACM Press.

Chervenski, P. and Ryan, S. (2014). MultiNEAT. http://multineat.com, configuration file: http://pastebin.com/TTY3UKtt.

Dellaert, F. and Beer, R. D. (1996). A developmental model for the evolution of complete autonomous agents. In *From Animals to Animats 4: Proc. of the 4th International Conference on Simulation of Adaptive Behavior (SAB 1996)*, pages 393–401. MIT Press.

Devert, A., Bredeche, N., and Schoenauer, M. (2011). Robustness and the halting problem for multicellular artificial ontogeny. *Evolutionary Computation, IEEE Transactions on*, 15(3):387–404.

Eggenberger Hotz, P. (1997). Evolving morphologies of simulated 3D organisms based on differential gene expression. In *Proc. of the 4th European Conference on Artificial Life (ECAL 1997)*, pages 205–213. MIT Press.

Joachimczak, M., Suzuki, R., and Arita, T. (2014). Fine grained artificial development for body-controller coevolution of soft-bodied animats. In *Artificial Life 14: Proc. of the 14th International Conference on the Synthesis and Simulation of Living Systems*, pages 239–246. The MIT Press.

Joachimczak, M. and Wróbel, B. (2012). Co-evolution of morphology and control of soft-bodied multicellular animats. In *Proc. of the 14th International Conference on Genetic and Evolutionary Computation*, GECCO '12, pages 561–568. ACM.

Komosinski, M. and Rotaru-Varga, A. (2002). Comparison of different genotype encodings for simulated three-dimensional agents. *Artif. Life*, 7(4):395–418.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evol. Comput.*, 19(2):189–223.

Pilat, M. L., Ito, T., Suzuki, R., and Arita, T. (2012). Evolution of virtual creature foraging in a physical environment. In *Artificial Life XIII: Proc. of the 13th International Conference on the Simulation and Synthesis of Living Systems*, pages 423–430. MIT Press.

Rubenstein, M., Cornejo, A., and Nagpal, R. (2014). Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799.

Schramm, L. and Sendhoff, B. (2011). An animat's cell doctrine. In *ECAL 2011: Proc. of the 11th European Conference on the Synthesis and Simulation of Living Systems*, pages 739–746. MIT Press.

Sfakiotakis, M. and Tsakiris, D. P. (2006). Simuun : A simulation environment for undulatory locomotion. *Int. J. Model. Simul.*, 26:350–358.

Sims, K. (1994). Evolving virtual creatures. In *Proc. of the 21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 15–22. ACM Press.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evol. Comput.*, 10(2):99–127.

Tufte, G. (2011). *Metamorphosis and Artificial Development: An Abstract Approach to Functionality*, volume 5777, chapter 11, pages 83–90. Springer Berlin Heidelberg.

Partial Redundancy and Morphological Homeostasis: Reliable Development through Overlapping Mechanisms

Micah Brodsky[1]

[1]Massachusetts Institute of Technology, Cambridge, MA 02139
micahbro@csail.mit.edu

Abstract

How might organisms grow into their desired physical forms in spite of environmental and genetic variation? How do they maintain this form in spite of physical insults? I show how these questions may have a common answer, a process of morphological homeostasis built from overlapping, partially redundant mechanisms.

Introduction

The physical forms of multicellular organisms are amazingly robust, developing correctly in spite of substantial environmental and genetic variation. This phenomenon was dubbed the "canalization" of development by Waddington (1942), reflecting the notion that there seems to exist some sort of restoring force pulling developing organisms back to their expected phenotype whenever perturbed. The most dramatic example may span entire phyla, as organisms within a single phylum start from dramatically different initial conditions yet converge to a common "phylotypic" stage of development, before differentiating into their characteristic larval forms (Kirschner and Gerhart, 2005). Similar convergence effects in spite of environmental perturbations can also be seen to varying degrees in the adult forms of animals, ranging from wound healing, to limb regeneration, to complete body reassembly after disaggregation, as in the hydra (Gierer et al., 1972).

What sorts of principles and tools does nature employ to produce such astonishing robustness? Can we master them ourselves, whether for engineering robust systems or for a deeper understanding of natural phenomena?

Morphological Homeostasis

Waddington's hypothetical "restoring force" of development cannot be completely hypothetical. For the dynamics of a physical system, such as an organism, to converge to a common attractor, the dynamics must be sensitive to the present state of the system – there must be feedback. Though such sensitivity can be a natural consequence of inanimate dynamics, for example, the surface tension that draws a droplet into a sphere, the complexity of biological forms strongly suggests explicit feedback control – an idea explored in this paper. We might dub Waddington's phenomenon, as extended to the adult, "morphological homeostasis".

Redundancy and Partial Redundancy

Is feedback control as an organizing principle enough to explain the reliability of development? In engineered systems, high reliability is typically achieved through redundancy, not feedback. For maintaining homeostasis, however, redundancy brings hazards of its own.

Consider an example from engineering, the RAID-5 disk array. Such a system uses $n + 1$ hard drives to provide n drives' worth of space. Through clever use of parity bits, it can survive a single drive failure with no loss of data or availability and with negligible performance degradation. Unfortunately, a common consequence is that the first drive failure goes completely unnoticed, until the second drive fails some time later and the entire data set is lost.

Full redundancy has a fundamental flaw: because it is so successful in hiding early failures, necessary steps to restore the system to its original, healthy state are neglected. Such systems are vulnerable to invisible deterioration: "rot". Redundancy becomes an expendable resource, a finite buffer against damage that, once depleted, is gone for good.

Nature has long since explored the design trade-offs here. The excessive kidney capacity we are born with is a good example, likely an acceptable compromise given our limited lifespans. On the other hand, for the integrity of our genes, such expendable redundancy is completely inadequate.

Animal genomes incorporate an entire hierarchy of redundancy measures. At the lowest level is a form of full redundancy reminiscent of RAID-1: the two-way mirroring within the double-stranded DNA polymer. There is, however, a crucial difference: in a cell, regular maintenance is tightly coupled into the system, not an afterthought to be handled by some outside process (i.e., a harried system administrator). Like in aviation, a cell that fails inspection is "grounded" – it enters senescence, ceasing to divide, or undergoes apoptosis, removing itself from the system. Moreover, a cell doesn't need outside inspectors carefully following a maintenance protocol; it inspects and repairs itself. Such intrinsic self-maintenance is a significant improvement over the blind redundancy furnished by a hard drive array.

Atop this 2-way mirror set lie multiple layers of further redundancy, but none so rigid and symmetrical; not replication, but imperfect redundancy, where the replicas are only similar at best and sometimes very different. A striking example is the diploid structure of animal cells: two nearly-complete but non-identical copies of the program code are included and executed simultaneously. Identicality (homozygosity), indeed, is often downright harmful.

Genes are also duplicated throughout the genome, and rarely are the copies identical. Unless there is a selective advantage to increased RNA throughput through simultaneous transcription (as in the unusual case of ribosomal RNAs, for example), identical duplicates constitute expendable redundancy: so long as accidental damage to a gene is more likely than successful re-duplication, spare copies are likely to be lost through neutral drift. Instead, over evolutionary time, any accidental copies that remain diverge and acquire new functions. Much of the original functionality remains, to the extent that a knock-out of either copy is often survivable, but not without some cost.

Most extreme is the case when redundancy is provided by completely unrelated genetic components through differing physical processes. Animal physiology is rife with highly divergent mechanisms converging on a common purpose. For example, blood loss at a wound is held in check simultaneously by the platelet clotting system, the thrombin/fibrinogen clotting system, and vasoconstriction. Why so many complex mechanisms? Why are these not pared down through neutral drift? In spite of their overlap, each independent mechanism seems to confer its own, unique selective advantage. That is, the mechanisms are not fully redundant, they are *partially* redundant. Damage to one is disadvantageous (e.g. as a hemophilia), but survivable.

Why should nature prefer partial redundancy? Why not do things one way and do it well? As a form of redundancy, of course, partial redundancy offers a buffer against damage and stress, bodily, genetic, or environmental. Unlike full redundancy, however, a component lost or weakened will cause detectable degradation. The gaps in redundancy are visible, and precisely *because* they are visible, partial redundancy provides feedback – feedback that favors regeneration (either somatic or selective), or even learned avoidance of danger. Partial redundancy, much more than full redundancy, facilitates homeostasis.

This paper explores a detailed case study in partial redundancy, arising in the problem of morphological homeostasis: how an organism attains and maintains its physical form, in spite of external insults, environmental variation, and internal evolutionary changes. The physical substrate used is the deformable surface model of Brodsky (2014c, Ch. 2), a rich, 2.5-dimensional physics that caricaturizes the mechanics of embryonic epithelial tissue. Taming this physics requires a fair amount of new mechanism for sensing and for actuation. In the course of developing this mechanism, the need for partial redundancy arises naturally. The remarkably robust results, evaluated informally, show how effective homeostasis through partial redundancy can be.

Model Background

On stylized, deterministic substrates like cellular automata, the value of feedback and redundancy is not so apparent, but deep challenges surface when a model's physics become sufficiently rich. With sophisticated mechanics, available strategies for development become more varied, but also, their effects less predictable, less modular.

Recent years have seen the use of increasingly sophisticated physical models (e.g. Chen and Brodland (2008); Disset et al. (2014); Doursat et al. (2012)). The model employed here (Brodsky, 2014c, Ch. 2), unlike the more common mass-spring models, improves mechanical richness by specializing to epithelial (sheet-like) tissues. This work also focuses on development by embryomorphic mechanical transformations, not by cell proliferation. However, the concepts should be applicable to any rich, 3d physics where cells can sense and manipulate their mechanical environment.

The model here is a "vertex model", a representation of a foam-like sheet of polygonal, tightly adhering cells in terms of the positions of their vertices. Cell shapes, and hence vertex positions, are governed by surface tension and internal elasticity. The model is extended into 3d by the addition of flexural springs at each cell-cell junction, the spring constant determining stiffness.

Cells are regulated by simple software agents, the realization in terms of genes not a focus of this work. Cells can sense properties of their mechanical conformation such as elongation or curvature and can influence it through neighbor-neighbor tractions (an adjustment to edge tension) or by modifying set-points such as flexural angle. Sufficient traction or external force will cause cells to intercalate, rearrange, and flow. How these effects can be profitably applied is a key focus of the paper.

Decomposing the Problem

Natural biological structures are complicated, combining multiple subparts with differing characteristics. We can simplify the problem of engineering morphological homeostasis by breaking it into a cascade of two subproblems: patterning and actuation. Patterning – "what goes where" – consists of laying out a body plan for the structure. Actuation – "what happens here" – represents the processes of local mechanical transformation necessary to create the desired features, given a pre-existing global body plan. Of course, these problems are not independent – the global pattern affects how actuation efforts interact, and updates to the global pattern require updates to the local features. Similarly, local actuation alters the geometric properties of the substrate, modifying the patterning process, as well as rearranging already patterned cells. However, so long as the goals of patterning and actuation are compatible, I show that the combination of appropriately robust patterning and actuation algorithms can yield a robust and stable complete solution.

The presence of conserved compartment maps in animals, an invisible and highly conserved pattern of gene expression prior to detailed morphogenesis (Kirschner and Gerhart, 2005), suggests that nature may use a similar decomposition strategy. Since perturbations in early, pre-morphogenesis development as well as local injuries to the final form can heal, global patterning and local actuation are both likely to involve feedback mechanisms.

The first problem, body plan patterning, can be solved by a patterning mechanism that is robust to widely variable substrate geometries and produces meaningfully consistent patterns before and after deformation. The patterning mechanism must also self-correct in the face of perturbations, without requiring a clean slate restart; incremental corrections to pattern and geometry must eventually converge, after all. These requirements all but eliminate self-timed pre-patterning (Brodsky, 2014a), which cannot respond to unexpected deviations, and likely disfavor fixed-wavelength Turing-type mechanisms, which have a preferred body size and may reconfigure under deformation (although note Meinhardt (1993)). Morphogenetic fields with self-sustaining sources (e.g. as in Doursat et al. (2012)) might be usable, with some caveats due to geometry (Brodsky, 2014c, Ch. 4). However, the normal neighbors patterning mechanism of Brodsky (2014b,c, Ch. 4), where patterns are specified through a purely topological description (an adjacency graph) and maintained continuously – hence tolerating substantial distortion – fits almost perfectly.

The core of this case study, then, is devoted to the problem of "what happens here": how to produce and maintain simple geometric features in spite of perturbations. We have at our disposal several mechanical actuation mechanisms, including cell shape change, apico-basal constriction, and neighbor traction forces (for simplicity, I don't consider changes in cell number here). Producing geometric features using these mechanisms is not too hard, given a known initial state. However, given perturbations, the initial state is *not* known. Instead, we must find techniques that respond appropriately to the system's pre-existing state.

Sensitivity to the state of the system – feedback – requires either that the intrinsic physics of the system be sensitive to system state (e.g. mechanical restoring forces) or that explicit feedback sensors be deployed by the control algorithm. Geometric structure involves numerous degrees of freedom, many of which are uninteresting (e.g. the relative arrangement of equivalent cells) or undesirable (e.g. high-frequency Fourier components). It can be valuable to leave such degrees of freedom to autonomous energy-minimization dynamics, for example, viscous relaxation, avoiding the control algorithm having to treat them explicitly. On the other hand, certain degrees of freedom represent key control targets. For these, we require sensors.

Sensing Curvature

For our first attempt at controlling geometry, consider spherical curvature – to produce spherical caps of varying radii, and hence varying subtended angle (e.g. Figure 1). First, we need a distributed, scale-invariant measure of curvature, built from local sensors.

Classical local measures of spherical curvature, such as Gaussian curvature and mean curvature, are not scale-invariant but instead provide curvature radii; they indicate how tightly curved the surface is locally but not how much curvature the surface encompasses as a whole. Gaussian curvature can be integrated over area to produce a dimensionless invariant related to the subtended angle (by the Gauss-Bonnet theorem), but this is an extensive quantity. In general, measuring extensive quantities seems to require leader election or an equivalent broken symmetry (Brodsky, 2014c, Ch. 5). It would be preferable to avoid this complication.

Another approach is to consider global properties based on length and area. For example, on a spherical cap, the ratio of area to the square of some linear dimension (e.g. perimeter) uniquely identifies the angle subtended. Without a leader, area and perimeter may not be directly measurable. However, the ratio of area to perimeter is easily measured (the 2D analogue of surface area to volume ratio, inverted), providing a second non-scale-invariant measure of curvature. This can be combined with a local measure of curvature – for example, multiplying by average mean curvature – to produce a scale-invariant measure of global curvature.

By trial and error, I found an interesting variation that worked well. Rather than combining the ratio of area to perimeter with another global measure, I combined it with a purely local measure of curvature, producing a hybrid measure that is partly local, partly global. This mirrors the effects of actuation, also partly local, partly global. The measure I found most effective is the product of the area-perimeter ratio and the extrinsic radius of curvature along the axis parallel to the region boundary – that is, the local circumferential curvature. Such a cocktail is, interestingly, an example of sensor-level partial redundancy.

Actuating Curvature

Now that we have a sensor for curvature, we must build an actuator. How? As noted before, surfaces have numerous degrees of freedom; all of them need to be stable, and some of them need to reach particular control targets. In almost any representation, they are cross-coupled, due to the constraints of surface geometry and the complicated dynamics of deformation and flow.

For example, one might instruct each cell to bend itself in accordance with the sign of the error reported by the curvature sensor. Such "extrinsic" curvatures can be driven by, e.g., apical/basal constriction. This approach, however, suffers from two serious flaws: it is geometrically inconsistent, and it does nothing to keep undesirable degrees of freedom under control. It is inconsistent for the same reason one cannot flatten an orange peel without tearing it: extrinsic curvatures require, in general, non-Euclidean geometries within the surface. Distances between points within the surface must change in order to accommodate the extrinsic curvature. If a surface is deformed extrinsically, non-Euclidean "intrinsic curvature" will necessarily be generated by elastic deformation and plastic intercalation, at the cost of high stresses, which fight against the bending forces and often lead to buckling instabilities, oscillations, and worse.

For example, a small circular disc subject to uniform extrinsic bending will yield a spherical cap, but beyond a certain critical size, it will spontaneously buckle cylindrically; the spherical conformation becomes unstable. Ideally, plastic deformation would set in before buckling, and the equilibrium intrinsic curvature would relax toward a spherical configuration. This is difficult to achieve, however, requiring substrates that are plastically soft yet flexurally quite stiff, and the high stresses involved remain a liability.

The complementary strategy, actuating on intrinsic curvature, is similarly geometrically inconsistent but has some notable properties. Unlike extrinsic curvature, which cells can directly manipulate, the relationship between what a cell can do locally and the resulting effects on intrinsic curvature is quite nontrivial (given by the Brioschi formula). Small changes to curvature can be produced by each cell changing its size and shape – adjusting its aspect ratio, for example. The effect on curvature is then a function of the differences in changes expressed among nearby cells. However, large changes must be achieved by plastically rearranging cells rather than simply distorting them, lest we demand that cells flatten into pancakes or stretch into spaghetti. A more useful actuator for large intrinsic curvatures is thus cell-cell traction, by which cells can intercalate with their neighbors.

How should cells exert traction forces in order to produce a given curvature? This is complicated. For the case of axisymmetric curvature, however, as in a spherical cap, the "purse string" strategy is an option: if curvature is too small, cells near the edge should pull on their circumferential neighbors, so as to shrink the mouth of the region. If curvature is too large, cells should pull on their radial neighbors, so as to enlarge it.

This sort of boundary-focused purse-string traction can be orchestrated, for example, by having the boundary emit a decaying gradient proportional in strength to the locally reported curvature error. The shape of the gradient then informs cells which direction and how hard to pull on their neighbors. The simplest approach might be to derive the orientation from the gradient vector or the level curves (choosing depending on the sign), and this works. I used an alternative source, the principal axes of the Hessian (negative axis along the boundary, due to sources, and positive axis elsewhere), which seemed slightly more effective.[1]

The effects of such purse-string traction are several. The application of traction forces leads to net stresses and bending moments in the surface, tending to open up or close the mouth of the region, as intended. In response, cells intercalate as expected, circumferentially or radially, leading to changes in intrinsic curvature. However, so long as curvature error persists, the rearrangement is incessant. Reorienting after each rearrangement, cells continue to grapple on one another, rearranging repeatedly. This continuing churn nullifies the yield strength of the cellular lattice and leads to viscous-like relaxation, which is both an asset and a liability. Churn relaxation is helpful because, as alluded to previously, it provides a natural mechanism for uninteresting and undesired degrees of freedom to relax and stabilize, without explicit control. It is problematic because the desired target degrees of freedom relax as well, making it difficult to sustain more than small deformations.[2]

The complementary problems exhibited by extrinsic bending and purse-string traction suggest that their combination might be more successful than either in isolation. Indeed, merely running them simultaneously, with no coordination, produces a drastic improvement. The combination of purse string traction as above and an integral controller on extrinsic bending, both using the same curvature feedback sensor, yields a stable and robust algorithm for producing spherical caps of arbitrary desired curvature. Figure 1 shows this tandem actuation mechanism in action, illustrating the results for several different target values of curvature.

At first glance, one might expect that the two actuation mechanisms ought to be tightly correlated, so that consistent intrinsic and extrinsic curvatures would be produced. However, the precise combination turns out to be quite forgiving. As the integral controller governing extrinsic bending ratchets up, intrinsic churn relaxation begins to lead towards rather than away from the desired equilibrium. At the same time, as cells rearrange, both autonomously and deliberately, the stresses generated by inconsistent curvatures are relaxed. Indeed, even without any coherent direction at all to the traction forces – a traction random walk – the combination of traction and extrinsic bending is sufficient. Convergence is

[1] Note that such actuation profiles are not scale-invariant, due to the fixed characteristic length scale of the gradient's decay. However, because the feedback sensors are scale-invariant, the resulting control algorithm is still quite flexible across a range of scales.

[2] There is also a subtle mathematical limitation to purse-string traction and other intrinsic actuation methods: they become singular when the surface is flat. Starting from a flat conformation, purse-string traction is weak and has no way to influence which way the surface will buckle. The sign of its influence depends on the sign of the existing extrinsic curvature.

slower and stresses are higher, but it works. In general, the relative calibration of intrinsic and extrinsic control affects the time to convergence and the stress profile, but the ultimate equilibrium is robust.

Complex Structures from Simple Pieces

Now that we have the beginnings of an understanding of geometric control for simple, homogeneous regions, how might we proceed to more complicated structures? Rather than developing a slew of more complicated sensors, actuators, and controllers, each with multiple degrees of freedom, it would be simpler if we could instead assemble multiple elementary features along a body plan pattern, each feature region running some simple control law. With actuation controllers like our example above, however, simply cutting and pasting regions together does not work well. Controllers must behave compatibly along shared boundaries, or they will fight each other. Even if curvatures can be carefully selected to match up, evolvability is impaired, because further revisions will require consistent modifications in multiple places simultaneously.

Instead of directly coupling tightly controlled components to each other, a better strategy might be to connect them through special combiner regions (or "combinators", to borrow a term from computer science) – a special type of actuation controller that furnishes sort of weakly controlled glue to couple otherwise incompatible boundaries together. Instead of tightly specifying all properties of the structure, one could specify only certain key regions and features, relying on combiner regions to interpolate between them for the remainder. Such combiner regions would insulate individual components from the geometrical and mechanical side effects of other components, allowing their controllers to operate quasi-independently.

Through the principle of relaxation, simple combiners are constructed easily. For small structures, I found that no active controller is needed, just a routine to ensure cells are reset their default properties. The churn injected from the jostling of neighboring regions' actuators is enough to cause mechanical relaxation, producing smooth connector regions with minimal curvature. For larger structures, it's necessary to add a controller that deliberately relaxes the surface through cell-cell traction; a simple random walk of traction will often suffice. A more aggressive approach might use a smoothing geometric flow (e.g. exerting traction along the major axis of the Hessian of Gaussian curvature).

By definition, a weakly controlled relaxation combiner does little to dictate the relative positions of the regions it connects, beyond the topological constraints imposed by the body plan. Where, then, do the regions end up? The body plan patterning mechanism may initially lay out the connected regions in some predictable fashion, but they effectively "float" upon the combiner, and in the long run, they move to occupy positions that minimize mechanical energy. Typically, this process is dominated by the bending energy. Regions can be modeled, in a sense, as interacting by virtual forces, dependent on their curvatures. Regions of the same sign of curvature typically repel, while those of opposite sign attract. If the global conformation leads to the formation of a bend in the combiner region, subsidiary regions may interact with this curvature as well. For example, when several regions of positive curvature float within a spherical combiner, they frequently align themselves along a circumferential ring, like spokes of a wheel (e.g. Figure 2a). Such virtual forces can often be relied on to produce a particular, final conformation in space.

Figure 2 shows a few examples of this approach, where independently controlled lobes are arranged by virtue of their interaction forces within a relaxation combiner. The number of lobes, their sizes and curvatures, and the divisions of the combiner can all be independently specified. However, there is no direct control available over the relative positions of the lobes. Even breaking the lobes into groups under different combiners does not meaningfully affect their positions (see Figure 2b); pure relaxation combiners are, to a good approximation, fully associative.

A more sophisticated combiner might manipulate the layout of its subsidiary regions by adding deliberate tractions and bending moments so as to customize the virtual interaction forces. More simply, however, we can break the associativity of the combiners with additional passive forces and use the resulting non-associative combiners to produce more complex shapes. An easy way to do this is with differential adhesion, such that different combiners have mutual disaffinity and hence are shaped by the surface tension forces along their boundaries. Figure 3 shows several examples of structures grown this way.

Evaluation

In spite of our meager toolbox consisting of one control law and two closely related combiners, the variety of structures we can declaratively produce is beginning to get interesting. It remains to be shown that the structures exhibit the robustness properties I have claimed, including self-repair, approximate scale invariance, and tolerance of unexpected parameter variations.

Geometric self-repair follows easily from the feedback control mechanism. One can even take geometric results of running under one program, switch to a different program, and watch the structure reform. The results are indistinguishable from structures produced starting with a sphere.

Approximate scale invariance can be demonstrated by running the same program on different size domains. Figure 4 demonstrates the program of Figure 3a running on different sized substrates. Using the same set of parameters as before, originally tuned for the middle size (400 cells), the small size (192 cells) works perfectly. The large size (900 cells) has a tendency to twin lobes but otherwise converges

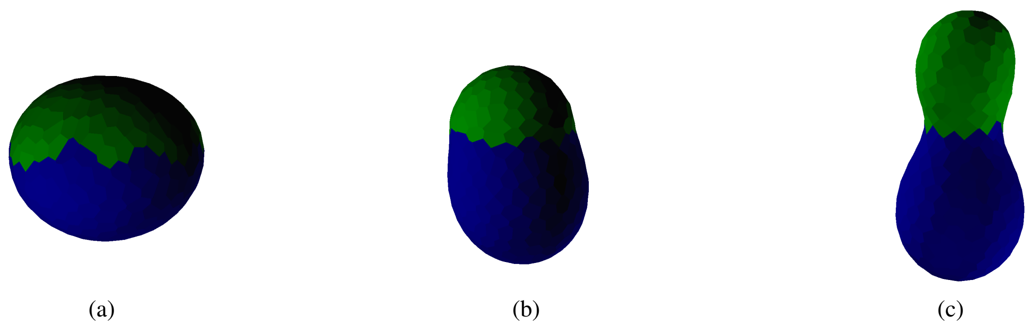


Figure 1: Lobes with controlled curvature – spherical surfaces divided into two regions (via normal neighbors), where green pursues a target curvature using purse-string traction and extrinsic bending, while blue relaxes passively (see section ‘Complex Structures from Simple Pieces’). Three different target curvatures are illustrated, with ratios $1 : 3 : 5$ respectively.

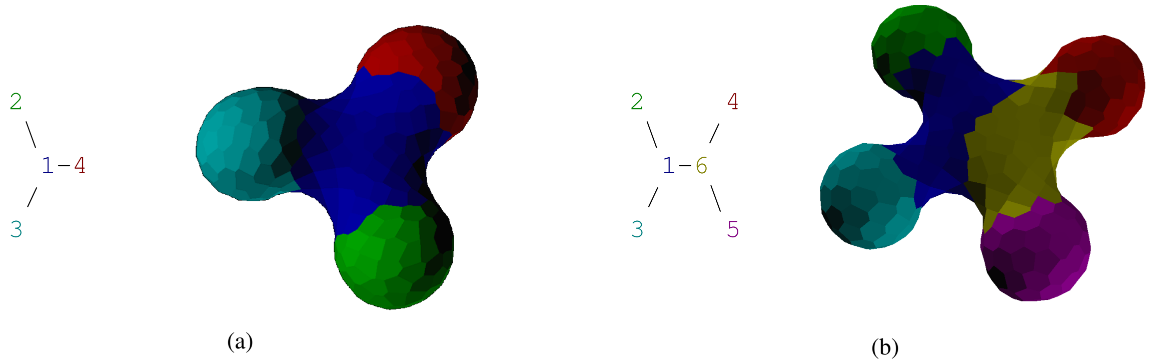


Figure 2: Simple compound structures and their associated normal neighbors body plans: (a) 3-lobe structure where red, green, and cyan regions control curvature while blue combiner region relaxes geometry. (b) 4-lobe structure where the lobes are split across two combiners (yellow and blue).

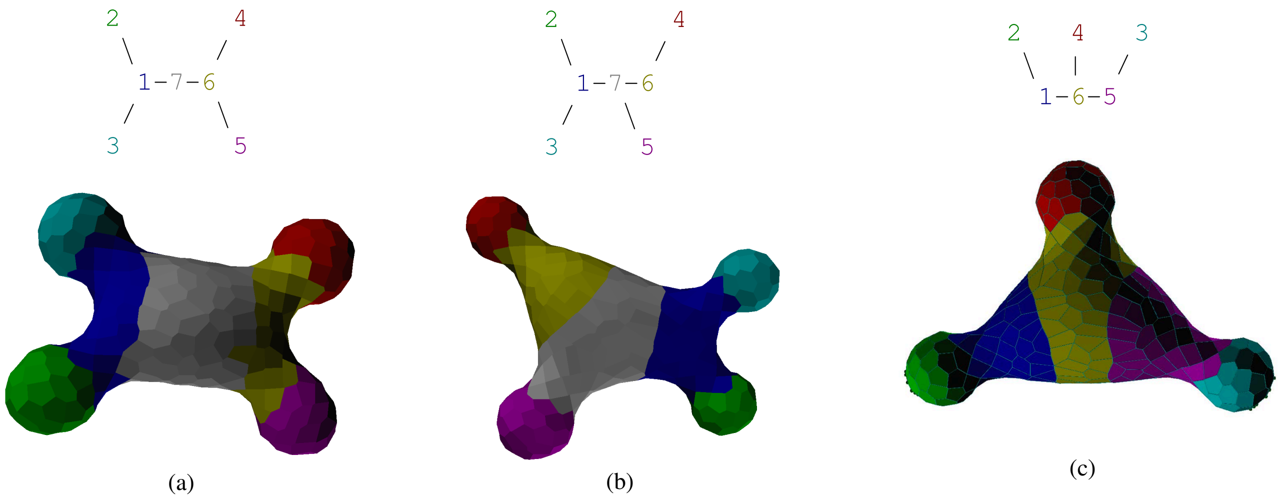
well (Figure 4b).[3] Such twinning can be avoided with adjustments to the body plan patterning algorithm, trading off speed for better convergence (e.g. Figure 4c, where the quorum sense morphogens have been configured to persist over longer distances) or simply tuning for the larger size (e.g. by increasing the “temperature” parameter).

The most interesting case to explore is that of unexpected parameter variation. For this purpose, I vary the stiffness of the substrate. This also affords the opportunity to explore the relative roles of the two actuation mechanisms in tandem. When substrates are stiffer, one should expect the extrinsic actuation to be more powerful, while on softer substrates, intrinsic actuation should be stronger.

Table 1 summarizes the results of informal trials under several different values of bending stiffness constant k_B and with several different “knockout variants” of the curvature control algorithm. As claimed, only mechanisms that combine both extrinsic bending and traction able to succeed in all cases (and at all with the middle stiffness). A lack of *directed* traction is a hindrance, but only inasmuch as it reduces the speed of convergence. Interestingly, there are cases where each of the other mechanisms still succeed. With high stiffness, one hardly notices the total loss of traction. With low stiffness, some patterns develop successfully even without bending (although their precise shapes are visibly altered).

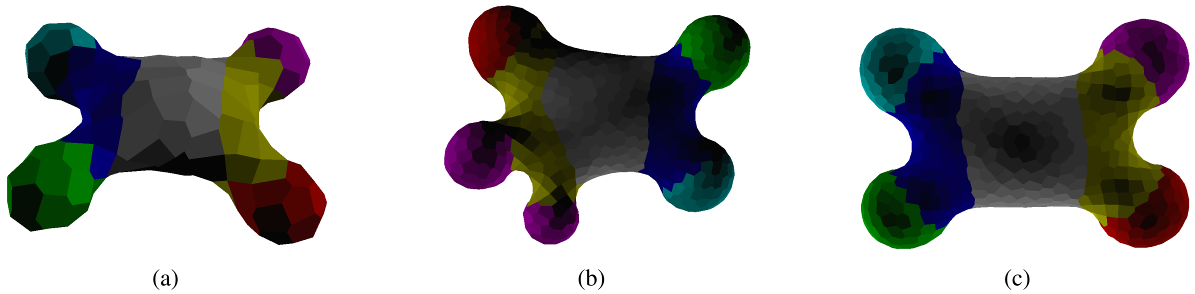
The tandem actuation mechanism thus exhibits partial redundancy: for many situations, multiple overlapping mechanisms are available, such that reduced function or complete failure of one pathway is quite survivable. However, due to the physical constraints of the problem, employing the full complement of mechanisms is often still helpful and sometimes absolutely necessary. The resulting combination mechanism is quite robust but irregularly so, giving confusing and seemingly contradictory results to knock-out experiments: Is the bending pathway necessary for curvature development? Is the traction pathway necessary for cur-

[3] In fact, the large size develops with extensive temporary twinning showing fully-actuated curvature, which only resolves through churn and domain wall migration. The highly curved lobe regions have a particular tendency to remain twinned, in spite of the body plan, probably due to the influence of their mutual mechanical repulsion.



(a) (b) (c)

Figure 3: Compound structures using relaxation combiners with associativity broken by surface tension. Leaf nodes control curvature; non-leaf nodes are combiners. Combiner cells have adhesive self-affinity and mutual disaffinity such that internal edge tension is reduced and mutual edge tension increased by (a) 40% and (b), (c) 80%. (The stronger surface tension in the latter two helps produce more distinct conical features.) Pattern regions are of unequal size in part due to deliberate adjustments to quorum feedback (k_q) – halved in leaf nodes, and, in (a), doubled in region 7.



(a) (b) (c)

Figure 4: Program of Figure 3a running on different domain sizes. (a) Small, 192-cell domain. (b) Large, 900-cell domain, showing typical twinning that fails to resolve. (c) Large, 900-cell domain that avoids twinning via 10x reduction in decay rate of pattern region quorum sense morphogens.

vature development? Is the gradient field that directs traction necessary for curvature development? Differing conditions may produce differing "answers" to these questions. The situation is surprisingly reminiscent of the difficulties encountered in knockout experiments on real, live organisms (Lazebnik, 2002).

Discussion: Partial Redundancy

The canonical benefit of partial redundancy is resistance to rot, hidden degradation. The more "verbose" a system – more details in its specification, more parts in its realization – the more vulnerable it is to random damage. Yet, the above example, combining two partially redundant mechanisms, fared quite well under both knock-out damage and mechanical disruption. By making degradation visible and detrimental yet survivable, partial redundancy facilitates homeostasis – both at the genetic level, through cross-over and selection, and at the somatic level, through regeneration – making complex and verbose systems sustainable.

The benefits run deeper. The example here showed how to use partial redundancy as a weapon to attack a messy, hard-to-characterize system. Neither mechanism alone furnished an exact solution, but each was able to cover for the other's bugs and limitations. Even when both mechanisms were not essential, performance was better with both, if for no other reason than that the total force could be increased by combining multiple modes of actuation.

Partial redundancy also facilitates modularity: redundant mechanisms may be repurposed to new functions, and stresses placed on the system by changes are more easily absorbed. Companion techniques, such as passive homeostasis by relaxation (as in combiners), further help to neutralize

	$k_B = 0.8$	$k_B = 2.4$	$k_B = 8$
Tandem actuation	Default: Fail[a] Reduced limit stops: Ok	Ok	Ok
Bending only	Marginal (slow; complex patterns fail)[b]	Marginal (slow; high failure rate)[b]	Ok
Traction only	Marginal (complex patterns are slow or unsuccessful)[c]	Unsuccessful[c]	Unsuccessful[c]
Bending + random traction	Slow	Slow	Ok

[a] Fails by a lobe pinching off. I hypothesize this is due to excessively strong actuation collapsing the base of the lobe rather than allowing sufficient time for the main combiner body to slowly relax. Pinch-off can be prevented by putting tighter limit stops on actuation of *either* bending angle or traction strength. The latter gives somewhat more consistent shapes.

[b] Fails through the development of tight, hyperbolic creases.

[c] Unsuccessful cases never produce definitive lobes; only slight curvatures form.

Table 1: Summary of results with varying substrate bending stiffness k_B for default algorithm and several "knockout" variants.

cross-interactions between components. Partial redundancy thus fosters exploration.

Conclusions and Future Work

This paper explored development and regeneration as single framework, morphological homeostasis via explicit feedback control. Focusing on mechanical remodeling rather than cell proliferation, several techniques were proposed. Ultimately, no one technique was best; instead, partially redundant combinations were fastest and most robust. Complex structures were then produced by introducing "combiners", using passive relaxation to decouple key features. A unifying theme, with applications to both biomimetic design and developmental theory, was partial redundancy and the feedback it entails. Still, many questions remain.

The pinch-off pathology, briefly mentioned in Table 1, represents a larger problem only crudely addressed: substrates have limits, beyond which they fail. Actuation must be careful not to exceed these limits, or it will destroy its own substrate. The solution used here, enforcing fixed bounds on actuator outputs, is crude both because it is hand-tuned and because it may unnecessarily limit outputs (and hence speed and control authority) even where there is no imminent danger of damage. A more elegant mechanism might be for the substrate to recognize its own limits and express "pain" when over-exerted, causing actuation to back off (Beal, 2010).

A significant limitation with the approach in this paper is that all patterning happens simultaneously in a single stage, which is both biologically unrealistic and limits the amount of complexity that can be implemented without getting stuck in local minima. Hierarchical and cascaded patterning would alleviate this limitation, but how can such sequential mechanisms be reconciled with regeneration? The answer is not clear; perhaps backtracking is involved.

The strategy of partial redundancy is not limited to physical or biological systems. For example, multiple versions of a software library might, like chromosomes, run in tandem. Different, loosely-coupled mechanisms might cooperate to ensure system homeostasis and sustainable resource usage. Virtual "pain" mechanisms might restrain over-taxing activities. The possibilities for bio-mimetic software systems are wide open.

References

Beal, J. (2010). Functional blueprints: An approach to modularity in grown systems. In *International Conference on Swarm Intelligence*.

Brodsky, M. Z. (2014a). Self-timed patterning. In *7th International Workshop on Spatial Computing (SCW 2014)*.

Brodsky, M. Z. (2014b). Spatial patterning with the rule of normal neighbors (ext. abstract). In Sayama, H. et al., editors, *Artificial Life 14*, pages 817–818. MIT Press.

Brodsky, M. Z. (2014c). *Synthetic Morphogenesis: Space, time, and deformation.* PhD thesis, MIT.

Chen, X. and Brodland, G. W. (2008). Multi-scale finite element modeling allows the mechanics of amphibian neurulation to be elucidated. *Physical Biology*, 5(1):015003.

Disset, J., Cussat-Blanc, S., and Duthen, Y. (2014). Self-organization of symbiotic multicellular structures. In Sayama, H. et al., editors, *Artificial Life 14*, pages 541–548. MIT Press.

Doursat, R. et al. (2012). Embryomorphic engineering: Emergent innovation through evolutionary development. In *Morphogenetic engineering: toward programmable complex systems*, pages 275–311. Springer.

Gierer, A. et al. (1972). Regeneration of hydra from reaggregated cells. *Nature/New Biology*, 239(88):98–101.

Kirschner, M. W. and Gerhart, J. C. (2005). *The Plausibility of Life: Resolving Darwin's Dilemma*. Yale University Press.

Lazebnik, Y. (2002). Can a biologist fix a radio? Or, what I learned while studying apoptosis. *Cancer Cell*, 2(3):179 – 182.

Meinhardt, H. (1993). A model for pattern-formation of hypostome, tentacles, and foot in hydra: how to form structures close to each other, how to form them at a distance. *Developmental Biology*, 157:321–333.

Waddington, C. H. (1942). Canalization of development and the inheritance of acquired characters. *Nature*, 150:563–565.

MecaCell: an Open-source Efficient Cellular Physics Engine

Jean Disset, Sylvain Cussat-Blanc and Yves Duthen

University of Toulouse, IRIT, CNRS - UMR5505
21 allée de Brienne, 31042 Toulouse, France
{disset, cussat, duthen}@irit.fr

Abstract

We present an open source physics engine specialised for multi-cellular artificial organisms simulations. It is computationally efficient in comparison to gas-based and finite element models and more realistic than standard mass-spring-damper systems.

Introduction

Morphogenetic engineering can often make good use of some biologically plausible improvements. Mechanics, in particular, are quite important to a certain group of bio-inspired artificial life experiments. The usual tradeoff in cell simulation being between accuracy and computational efficiency, we have developed a model specialised in cellular physics which aims to stay efficient while precise enough for most artificial life applications. It is based on an improved mass-spring-damper (MSD) system, the use of which is widespread in the literature, mainly because of its computational efficiency. The main limitation in MSD systems is the difficulty they have when taking into account uneven adhesive forces and global tensegrity of a simulated biological system while allowing the simulation of freely moving cellular clusters in a 3D environment (Joachimczak et al. (2013)). We improved on this model by adding adhesion and collision springs, and we took inspiration from Euler-Bernouilli beam theory to account for flexure, torsion and shear, which are only possible with standard MSD system (which natively only handle compression) by using complex topologies.

Physics model

In our current cellular model, a cell is represented by a 3D position, an orientation and an implicit surface. Each cell also has a mass, a radius, a stiffness and an adhesive strength (influencing connection length, resistance to traction, flexure, torsion and shear). Cells are linked two by two by a simplified elastic beam modeled by one compression spring that embeds two tendons per cells. One is used to simulate the flexure relative to the compression spring and the other the torsion. These forces allow for the development of linear structures with interesting tensegrity characteristics.

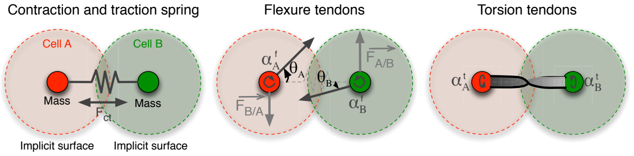


Figure 1: 3 different kinds of connections are used. Traction springs apply equal forces (of opposed directions) to cells A and B, torsion joint apply a torque and flexure joints applies both a torque and an orthogonal force to the opposed cell.

The contact surface between two cells is also used to determine the angular stiffness coefficients. These springs and articulations are created dynamically when the distance between two cells is under a given threshold (given by the cell properties) and are deleted when torn apart too strongly. An explicit integration scheme is used to update the world state.

Conclusion

A simplified 2D version of this engine has been successfully used with a developmental model that open-endedly grows virtual organisms in a constraintfull environment (Disset et al. (2014)) and the current version is being investigated as the basis for more complex 3D morphogenetic engineering experiments. Benchmarks have shown that our implementation of the model can run at 30 frames per seconds with about 20000 cells on a state of the art machine. We plan to test the engine against real in-vitro biological experiments. An implementation as well as videos are freely available at https://github.com/jdisset/MecaCell.

References

Disset, J., Cussat-Blanc, S., and Duthen, Y. (2014). Self-organization of symbiotic multicellular structures. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 3–5.

Joachimczak, M., Kowaliw, T., Doursat, R., and Wrobel, B. (2013). Controlling development and chemotaxis of soft-bodied multicellular animats with the same gene regulatory network. In *Advances in Artificial Life, ECAL*, volume 12, pages 454–461.

Musculo-Skeletal Models as Tools to Quantify Embodiment

Daniel F. B. Haeufle[1], Michael Günther[1], and Syn Schmitt[1]

[1]University of Stuttgart, 70569 Stuttgart, Germany
daniel.haeufle@inspo.uni-stuttgart.de

Musculo-skeletal models

Human and animal movement is absolutely fascinating and can hardly be mimicked by technical devices, so far. It has been proposed that part of the movement generation and control can be associated to the non-linear characteristics of the bio-mechanical structures and morphology. Terms like *morphological computation* (Paul, 2006) and *intelligence by mechanics* (Blickhan et al., 2007) capture this idea.

Musculo-skeletal models allow to synthesize biological movements based on models representing the non-linear characteristics of muscles, tendons, ligaments, and other bio-mechanical structures. This approach allows for studying the contribution of these structures and, thus, their role in the control of human and animal movements.

Quantifying control effort with information entropy

In a recent study (Haeufle et al., 2014), we showed that the control effort of periodic hopping is reduced by the muscle contraction dynamics (32 Bits) as compared to DC-motor characteristics (660 Bits). To quantify control effort we proposed to measure the minimal information $I_{\min}$ required to control hopping. The information transmitted from a digital sensor to an actuator (muscle, DC-motor) can be quantified based on Shannon's information entropy. In a typical technical implementation, the controller assumes equal probability $p_i = 1/n$ of each of the n possible sensor values when digitizing a physical signal u. With this assumption, the total information transmitted with a signal is $I = \frac{T}{\Delta t} \log_2 \left(1 + \frac{u_{\max} - u_{\min}}{\Delta u}\right)$, where $u_{\min} \leq u \leq u_{\max}$ is the range of the sensor, Δu its amplitude resolution, Δt the temporal resolution, and T the total duration of the movement. The minimally required information $I_{\min}$ for a given actuator and controller was found by varying the parameters Δu and Δt under the constraint that hopping height and frequency were still in a biologically realistic range.

Similar to the quantification of *relevant information* introduced by Polani et al. (2006), we also limit the information transmitted to the actuator to identify the minimal information required for a given task. However, we focus on modifying the internal dynamics of the agent (muscle vs. DC-motor) and study the effect on $I_{\min}$ to ultimately study biological design, and, thus, embodiment. For a system generating hopping with coarse signal resolution and therefore little processed information, the control effort is low and the control is simple.

New perspective

The musculo-skeletal models developed in our group represent the physiological and morphological structure of biological agents in a level of detail which is adequate for studying typical movements. It is possible to vary the physiological and morphological structures and properties and, hence, to study their role in movement control. We think that this approach opens up unique possibilities to quantify embodiment and even compare it to technical systems.

Our approach to control effort quantifies the minimal information $I_{\min}$, which has to be processed to generate a specific movement with a specific controller. We hypothesize, that the differences in $I_{\min}$ found by varying and exchanging structures, e.g., actuators, give insights into the amount of embodiment. To further evaluate this idea, we would like to discuss our work in the context of other recently published approaches for the quantification of embodiment and morphological computation.

References

Blickhan, R., Seyfarth, A., Geyer, H., Grimmer, S., Wagner, H., and Günther, M. (2007). Intelligence by mechanics. *Philosophical Transactions of the Royal Society of London, Series A*, 365(1850):199–220.

Haeufle, D. F. B., Günther, M., Wunner, G., and Schmitt, S. (2014). Quantifying control effort of biological and technical movements: An information-entropy-based approach. *Physical Review E*, 89(1):012716.

Paul, C. (2006). Morphological computation: A basis for the analysis of morphology and control requirements. *Robotics and Autonomous Systems*, 54(8):619–630.

Polani, D., Nehaniv, C., Martinetz, T., and Kim, J. (2006). Relevant information in optimized persistence vs. progeny strategies. In *Artificial Life X*, pages 337–343. MIT Press.

Task Dynamics & the (Ecological) Information They Create

Andrew D Wilson[1] and Sabrina Golonka[1]

[1]Leeds Beckett University, Leeds UK
a.d.wilson@leedsbeckett.ac.uk

Embodied cognition is the hypothesis that behavior is not simply caused by the brain. Instead, behavior emerges from the interactions between brains in particular kinds of bodies embedded in environments that provide certain kinds of opportunities for activity. Theories of embodied cognition require a mechanism to support how these distributed resources can remain in contact with one another so that they can be assembled into task-specific solutions to problems. These theories of embodied cognition, especially the non-representational kinds, typically rely on James J. Gibson's (1979) notion of *ecological information* as the relevant mechanism, and there is extensive empirical support for the claim that this information both exists and is used by organisms to coordinate and control their activity.

For Gibson, information refers to structures in ambient energy arrays (e.g. light for vision) that are specific to the object or event in the world that caused the structure. This structure becomes information when we use it to coordinate and control our behavior, and this information supports what Gibson (1979) referred to as the *direct perception* of our environments.

This account of how we maintain psychological contact with the world has been formalized in the years since Gibson's death in the context of *task dynamics* (e.g. Saltzman & Kelso, 1987). Objects and events in the world can only be identified uniquely at the level of dynamics (the description of how a system changes over time, with reference to the forces causing the change; Bingham, 1995). Some of these dynamics can then be mapped into energy arrays as kinematic patterns (descriptions of change with no reference to underlying forces). These kinematic patterns, although not identical to the dynamical objects or events, can be specific to them (Runeson & Frykholm, 1983; Turvey, Shaw, Reed & Mace, 1981) and therefore be informative about them. Detecting a specifying kinematic pattern is equivalent to perceiving the underlying dynamic, and our perceptual systems are exquisitely sensitive to these patterns.

A given task dynamic can, by definition, only produce information about that particular dynamic. Information-based control of behavior is therefore task-specific, and so empirical research that investigates information-based explanations for behavior therefore follows four crucial steps (Wilson & Golonka, 2013; see there for specific research examples):

1. Identifying the task facing the organism at the level of task dynamics
2. Use this task dynamical analysis to identify a comprehensive list of the specific resources offered by the task to support a solution, in particular the task-specific information variables created by the task dynamics
3. Describe how these resources are assembled into a solution to the task at hand (generally a dynamical model built only from the resources identified in [2])
4. Test the model and it's predictions in real organisms

The purpose of this presentation is to introduce this ecological, biological notion of information and the related research programme to the artificial life community, where 'information' typically refers to measures of entropy, following Shannon. An information theoretic analysis may prove useful to our scientific understanding of ecological information but regardless, that is not the kind of information biological systems interact with as they perceive and act in their environments. That is Gibson's information. Understanding this ecological information is a critical part of understanding how biological life gets up to the things that it does. It may also, therefore, be of use to helping artificial life get up to those same things.

References

Bingham, G.P. (1995). Dynamics and the problem of visual event recognition. In Port, R. & T. van Gelder (eds.), *Mind as Motion: Dynamics, Behavior and Cognition*, (pp403-448). Cambridge, MA: MIT Press.

Gibson, J.J. (1979). *The ecological approach to visual perception*. Boston: Houghton Mifflin

Runeson, S., & Frykholm, G. (1983). Kinematic specification of dynamics as an informational basis for person and action perception: Expectation, gender recognition, and deceptive intention. *Journal of Experimental Psychology: General*, 112:617-632.

Saltzman, E., & Kelso, J. A. (1987). Skilled actions: a task-dynamic approach. *Psychological Review*, 94:84-106.

Turvey, M. T., Shaw, R. E., Reed, E. S., & Mace W. M. (1981). Ecological laws of perceiving and acting: In reply to Fodor and Pylyshyn (1981) *Cognition*, 9:237-304

Wilson. A. D., & Golonka, S (2013). Embodied cognition is not what you think it is. *Frontiers in Psychology*, 4:58. doi: 10.3389/fpsyg.2013.00058

Quantifying Morphological Computation based on an Information Decomposition of the Sensorimotor Loop

Keyan Ghazi-Zahedi[1], Johannes Rauh[2]
[1]Max Planck Institute for Mathematics in the Scienes, Inselstrasse 22, 04103 Leipzig, Germany
[2] Leibniz Universität Hannover, Welfengarten 1, 30167 Hannover, Germany
zahedi@mis.mpg.de, rauh@math.uni-hannover.de

Abstract

The question of how an agent is affected by its embodiment has attracted growing attention in recent years. A new field of artificial intelligence has emerged, which is based on the idea that intelligence cannot be understood without taking the embodiment into account. The contribution of an agent's embodiment to its behaviour is also known as morphological computation. In this work, we propose a quantification of morphological computation, which is based on an information decomposition of the sensorimotor loop into shared, unique and synergistic information. Using a simple model of the sensorimotor loop, we show that the unique information of the body with respect to the environment is a good measure for morphological computation.

Introduction

Morphological computation is discussed in various contexts, such as DNA computing and self-assembly (see Pfeifer et al., 2007c; Hauser et al., 2012, for an overview). In this publication, we are interested in quantifying morphological computation of embodied agents which are embedded in the sensorimotor loop. Morphological computation, in this context, is described as the trade-off between morphology and control (Pfeifer and Scheier, 1999), which means that a well-chosen morphology, if exploited, substantially reduces the amount of required control (Montúfar et al., 2014). Here, the term *morphology* refers to the agent's body, explicitly including all its physiological and physical properties (shape, sensors, actuators, friction, mass distribution, etc.) (Pfeifer, 2002). The consensus is that morphological computation is the contribution of the morphology and environment to a behaviour, that cannot be assigned to a nervous system or a controller. There are several examples from biology that demonstrate how the behaviour of an agent relies on the interaction of the body and environment. A nice example is given by Wootton (1992, see p. 188), who describes how "active muscular forces cannot entirely control the wing shape in flight. They can only interact dynamically with the aerodynamic and inertial forces that the wings experience and with the wing's own elasticity; the instantaneous results of these interactions are *essentially* determined by the architecture of the wing itself [. . .]"

One of the most cited example from the field of embodied artificial intelligence is the Passive Dynamic Walker by McGeer (1990). In this example, a two-legged walking machine preforms a naturally appealing walking behaviour, as a result of a well-chosen morphology and environment, without any need of control. There is simply no computation available and the walking behaviour is the result of the gravity, the slope of the ground and the specifics of the mechanical construction (weight and length of the body parts, deviation of the joints, etc.). If any parameter of the mechanics (morphology) or the slope (environment) is changed beyond a small threshold, the walking behaviour will not persist. In this context, we understand the exploitation of the body's and environment's physical properties as the embodiments effect on a behaviour.

It is important to note that talking about a system's behaviour in the context of embodied artificial intelligence does not make much sense if there is no agency involved. This means that the e.g. the Passive Dynamic Walker is basically nothing more than an interesting mechanical system. Yet, its purpose is to study how mechanical properties of the legs affect human walking, which is why it is often cited as an example for morphological computation. In the same way, the interaction of the physical properties of insect wings in the given example are meaningless, if there is no agent that exploits the properties to achieve a behaviour of interest (flying in this case of insects, walking in case of humans). In this paper, we focus on the quantification of body's physical interaction with it's environment, to which we refer as morphological computation. This does not mean that we are not aware that the flapping of the wings requires a control, i.e., that morphological computation needs to be induced by an agent. Discussing both aspects of morphological computation in detail is beyond the scope of this work, although they are both addressed by our information decomposition of the sensorimotor loop, as we discuss later.

Theoretical work on describing morphological computation in the context of embodied artificial intelligence has been conducted by (Hauser et al., 2011; Füchslin et al., 2012). In this publication, we study an information-theoretic approach to quantifying morphological computation which is based on two of our previous publications: In (Zahedi and Ay, 2013) we have investigated different quantifications of morphological computation, which all match the general intuition, but showed different results when applied to a simple model of the sensorimotor loop. In (Bertschinger et al., 2014) we have derived a general decomposition of a mutual information of three random variables into unique, shared, and synergistic information (Bertschinger et al., 2014). Here, we apply this information decomposition to the simple model of the sensorimotor loop in order to improve our previous measures of morphological computation.

The paper is organised in the following way. The next section discusses the sensorimotor loop and its representation as a causal graph. The third section describes the bivariate information decomposition from Bertschinger et al. (2014). Based on the information decomposition, the fourth section introduces the unique information as a measure for morphological computation in the sensorimotor loop. The fifth section presents numerical results, which are then discussed in the final section. An appendix explains how we computed our measure of morphological computation.

Sensorimotor Loop

Our information theoretic decomposition of the mutual information requires a formal representation of the sensorimotor loop, which is introduced in this section. In our understanding, a cognitive system consists of a brain or controller, which sends signals to the system's actuators, thereby affecting the system's environment. We prefer the notion of the system's *Umwelt* (von Uexkuell, 1934; Clark, 1996; Zahedi et al., 2010), which is the part of the system's environment that can be affected by the system and which itself affects the system. The state of the actuators and the *Umwelt* are not directly accessible to the cognitive system, but the loop is closed as information about the *Umwelt* and the body is provided to the controller through the sensors. In addition to this general concept of the sensorimotor loop, which is widely used in the embodied artificial intelligence community (see e.g. Pfeifer et al., 2007a) we introduce the notion of *world* and by that we mean the system's morphology and the system's *Umwelt*. We can now distinguish between the intrinsic and extrinsic perspective in this context. The world is everything that is extrinsic from the perspective of the cognitive system, whereas the controller, sensor and actuator signals are intrinsic to the system. This is analogous to the agent-environment distinction in the context of reinforcement learning (Sutton and Barto, 1998), in which the environment is understood as everything that cannot be controlled arbitrarily by the agent.

The distinction between intrinsic and extrinsic is also captured in the representation of the sensorimotor loop as a causal or Bayesian graph (see Fig. 1). For simplicity, we only discuss the sensorimotor loop for reactive systems. This is plausible, because behaviours which exploit the embodiment are usually better described as reactive and not as deliberative. The most prominent examples are locomotion behaviours, e.g. human walking, swimming, flying, etc., which are all well-modelled as reactive behaviours.

The random variables S, A, and W refer to sensor, actuator, and world state, and the directed edges reflect causal dependencies between the random variables (see Klyubin et al., 2004; Ay and Polani, 2008; Zahedi et al., 2010). Everything that is extrinsic is captured in the variable W, whereas S and A are intrinsic to the agent. The random variables S and A are not to be mistaken with the sensors and actuators. The variable S is the output of the sensors, which is available to the controller or brain, the action A is the input that the actuators take. Consider an artificial robotic system as an example. Then the sensor state S could be the pixel matrix delivered by some a sensor and the action A could be a numerical value that is taken by a motor controller to be converted in currents to drive a motor.

Throughout this work, capital letters (X, Y, $\dots$) denote random variables, non-capital letters (x, y, $\dots$) denote specific values that random variables can take, and calligraphic letters ($\mathcal{X}$, $\mathcal{Y}$, $\dots$) denote the alphabets for the random variables. For example, the random variable X may take the value $x \in \mathcal{X}$. Greek letters (α, β, $\dots$) refer to generative kernels, i.e. kernels which describe an actual underlying mechanism or a causal relation between random variables.

The random variables that we consider depend on time, which we model as a discrete parameter $t \in \mathbb{N}$. For example, the output of the sensors corresponds to a sequence of random variables $S_1, S_2, \dots$, with one random variable S_t for each time step t. We are mostly interested in what happens in a single time step. Therefore, we use the following notation. Random variables without any time index refer to some fixed time t and primed variables to time $t + 1$. For example, the two variables S, S' refer to S_t and S_{t+1}.

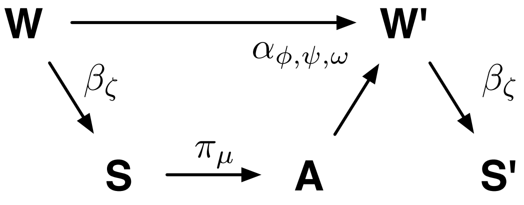


Figure 1: A formal model of the sensorimotor loop.

Formally, the sensorimotor loop is given by the probability distribution $p(w)$ and the kernels $\alpha(w'|w, a)$, $\beta(s|w)$, and $\pi(a|a)$, see Figure 1. We choose the same parameterisable binary model of the sensorimotor loop as in (Zahedi and Ay, 2013) (with an additional synergistic parameter, see Eq. (1)). It allows us to control the causal dependencies of

S, A, and W individually, and thereby enables us to evaluate the information decomposition in the sensorimotor loop and compare the result with our previous results. The model is given by the following set of equations:

$$\alpha_{\phi,\psi,\omega}(w'|w,a) = \frac{e^{\phi w'w+\psi w'a+\omega w'wa}}{\sum_{w''\in\Omega} e^{\phi w''w+\psi w''a+w''wa}} \tag{1}$$

$$\beta_\zeta(s|w) = \frac{e^{\zeta sw}}{\sum_{s''\in\Omega} e^{\zeta s''w}} \tag{2}$$

$$\pi_\mu(a|s) = \frac{e^{\mu as}}{\sum_{a'\in\Omega} e^{\mu a's}} \tag{3}$$

$$p_\tau(w) = \frac{e^{\tau w}}{\sum_{w''\in\Omega} e^{\tau w''}}, \tag{4}$$

where $a, w, s, w' \in \Omega = \{\pm 1\}$ and $\phi, \psi, \omega, \zeta, \mu, \tau \geq 0$. As in (Zahedi and Ay, 2013), we make the following two simplifying assumptions, which do not restrict generality too much. First, we assume that all world states $w \in \Omega$ occur with equal probability, i.e. $p(w = 1) = p(w = -1) = 1/2$. Second, we assume a deterministic sensor, i.e. $\zeta \gg 1 \Rightarrow p(s|w) = \delta_{sw}$, i.e. the sensor is a copy of the world state. The first assumption does not reduce generality much, because it only assures that the world state itself does not already encode some structure, which is propagated through the sensorimotor loop. The second assumption does not violate the generality of the model, because in a reactive system, the sensor state S and A can be reduced to a common state, with a new generative kernel $\gamma(a|w) = \sum_s \pi(a|s)\beta(s|w)$. Hence, keeping one of the two kernels deterministic and varying the other in the experiments below, does not reduce the validity of this model. This leaves four open parameters ψ, ϕ, ω, and μ, against which the morphological computation measure is validated.

Information decomposition is most often discussed in the context of binary logical functions such as XOR, and so it is useful to think of the one-step sensorimotor loop as such a logical function. The one-step sensorimotor loop is a simplified model of the causal diagram of the sensorimotor loop (which unfolds over time) that we did not present and discuss due to spacial constraints. A more thorough discussion can be found in (Zahedi et al., 2010; Zahedi and Ay, 2013; Pfeifer et al., 2007b). The one-step sensorimotor loop is sufficient to discuss, visualise and evaluate the information decomposition for the following reason. The decomposition and quantification of morphological computation is based on the joint distribution $p(w', w, a)$ which can be obtained e.g. by observation. An example for such an application would be the recording of an animal's motion with 3D motion capturing and vibromyography (muscle activity) sensors. The joint distribution $p(w', w, a)$ can then be obtained from the recorded data (which implicitly means that we assume ergodicity). The presented model (see Eqs. (1) to (4) and Fig. 1) is chosen such that it allows us to freely parametrise the joint distribution.

Information Decomposition

Next, we introduce the information decomposition that underlies our measure of morphological computation. We first explain this information decomposition in a general information theoretic setting and later explain how we use it in the sensorimotor loop.

Consider three random variables X, Y, Z. Suppose that a system wants to predict the value of the random variable X, but it can only access the information in Y or Z. How is the information that Y and Z carry about X distributed over Y and Z? In general, there may be *redundant* or *shared* information (information contained both Y and Z), but there may also be *unique* information (information contained in only one of Y or Z). Finally, there is also the possibility of *synergystic* or *complementary* information, i.e. information that is only available when Y and Z are taken together. The classical example for synergy is the XOR function: If Y and Z are binary random variables and if $X = Y \,\mathrm{XOR}\, Z$, then neither Y nor Z contain any information about X (in fact, X is independent of Y and X is independent of Z), but when Y and Z are taken together, they completely determine X (in particular, X is not independent from the pair (X, Y)).

The total information that (Y, Z) contains about X can be quantified by the mutual information $MI(X : (Y, Z))$. However, there is no canonical way to separate these different kinds of information. Mathematically, one would like to have four functions $SI(X : Y; Z)$ ("shared information"), $UI(X : Y\backslash Z)$ ("unique information of Y"), $UI(X : Z\backslash Y)$ ("unique information of Z"), $CI(X : Y; Z)$ ("complementary information") that satisfy

$$\begin{aligned} MI(X : (Y, Z)) = SI(X : Y; Z) + UI(X : Y \setminus Z) \\ + UI(X : Z \setminus Y) + CI(X : Y; Z). \end{aligned} \tag{5}$$

From the interpretation it is also natural to require

$$\begin{aligned} MI(X : Y) = SI(X : Y; Z) + UI(X : Y \setminus Z), \\ MI(X : Z) = SI(X : Y; Z) + UI(X : Z \setminus Y). \end{aligned} \tag{6}$$

A set of three functions SI, UI, and CI that satisfy (5) and (6) is called a *bivariate information decomposition* by Bertschinger et al. (2014). It follows from the defining equations and the chain rule of mutual information that an information decomposition always satisfies

$$MI(X : Y|Z) = UI(X : Y \setminus Z) + CI(X : Y; Z). \tag{7}$$

Equations (5) and (6) do not specify the functions SI, UI, and CI. Several different candidates have been proposed so far, for example by Williams and Beer (2010) and Harder et al. (2013). We will use the decomposition of Bertschinger et al. (2014) that is defined as follows[1]:

[1] The same functions were also proposed by Griffith and Koch (2014) starting from a measure for "union information" obtained from formal information-theoretic arguments.

Let Δ be the set of all possible joint distributions of X, Y, and Z. Fix an element $P \in \Delta$ (the "true" joint distribution of X, Y, and Z). Define

$$\begin{aligned}\Delta_P = \Big\{Q \in \Delta :& \\ &Q(X = x, Y = y) = P(X = x, Y = y) \\ &\text{and } Q(X = x, Z = z) = P(X = x, Z = z) \\ &\qquad \text{for all } x \in \mathcal{X}, y \in \mathcal{Y}, z \in \mathcal{Z}\Big\}\end{aligned}$$

as the set of all joint distributions which have the same marginal distributions on the pairs (X, Y) and (X, Z). Then

$$\begin{aligned}UI(X : Y \setminus Z) &= \min_{Q \in \Delta_P} MI_Q(X : Y|Z), \\ SI(X : Y; Z) &= \max_{Q \in \Delta_P} CoI_Q(X; Y; Z), \\ CI(X : Y; Z) &= MI(X : (Y, Z)) \\ &\quad - \min_{Q \in \Delta_P} MI_Q(X : (Y, Z)),\end{aligned}$$

where CoI denotes the interaction information (McGill, 1954), sometimes also called co-information. Here, a subscript Q in an information quantity means that the quantity is computed with respect to Q as the joint distribution.

One idea behind these functions is the following: Suppose that the joint distribution P of X, Y, and Z is not known, but that just the marginal distributions of the pairs (X, Y) and (X, Z) are known. This information is sufficient to characterize the set Δ_P, but we do not know which element of Δ_P is the true joint distribution. One can argue that the UI and SI should be constant on Δ_P; that is, shared information and unique information should depend only on the interaction of X and Y and the interaction of X and Z, but not on the threeway interaction.

The second property that characterizes the information decomposition is that the set Δ_P contains a distribution Q such that $CI_Q(X : Y; Z) = 0$. In other words, when only the marginal distributions of the pairs (X, Y) and (X, Z) are known, then we cannot know whether there is synergy or not. See (Bertschinger et al., 2014) for a more detailed justification and a proof how these properties determine the functions UI, SI, and CI.

In Bertschinger et al. (2014), the formulas for UI, CI, and SI are derived from considerations about decision problems in which the objective is to predict the outcome of X. Here, we want to apply the information decomposition in another setting: We will set $X = W'$, $Y = W$, and $Z = A$. In our setting, W and A not only have information about W', but they actually *control* W'. However, the situation is similar: In the sensorimotor loop, we also expect to find aspects of redundant, unique, and complementary influence of W and A on W'. Formally, since everything is defined probabilistically, we can still use the same functions UI, CI, and SI. We believe that the arguments behind the definition of UI, CI and SI remain valid in the setting of the sensorimotor loop where we need it. First, it is still plausible that unique and redundant contributions should only depend on the marginal distributions of the pairs (W, W') and (A, W'). Second, in order to decide whether W and A act synergistically, it does not suffice to know only these marginal distributions. Therefore, we believe that the functions UI, CI, and SI have a meaningful interpretation. In particular, we hope to be able to use the information decomposition in order to measure morphological computation. This view is supported by our results below, which indicate that the functions UI, CI and SI do indeed lead to a reasonable decomposition of $MI(W' : (A, W))$ and that the unique information $UI(W : W' \setminus A)$ is a reasonable measure of morphological computation, at least in our simple model of the sensorimotor loop.

The parameters of our model of the sensorimotor loop (Eqs (1) to (4)) can also be interpreted in terms of an information decomposition. Intuitively, ϕ corresponds to the unique influence of W on W', ψ corresponds to the unique influence of A on W', and ω corresponds to the complementary influence. However, the role of the other parameters ζ, μ, τ is less clear, and there is no clear correspondence for redundant information. The information decomposition has the advantage, that its definition does not depend on a parametrization. Note that if the "synergistic parameter" $\omega = 0$ vanishes, then it does not necessarily follow that $CI(W' : A; W) = 0$ (see Fig. 2). However, we do expect the complementary information to be small in this case.

Morphological computation

Morphological computation was described as the contribution of the embodiment to a behaviour. In our previous work, we derived two concepts to quantify morphological computation, which are both based on the world dynamics kernel $\alpha(w'|w, a)$.

The first concept assumes that the current action A has no influence on the next world state W', in which case the kernel $\alpha(w'|w, a)$ reduces to $\hat{\alpha}(w'|w)$. If this is the case, we would say that the systems shows maximal morphological computation, as the behaviour is completely determined by the world. To measure the amount of morphological computation present in a recorded behaviour, we calculated how much the data differed from the assumption by calculating the weighted Kullback-Leibler divergence $\sum_{w,a} p(w, a) D_{KL}(\alpha(w'|w, a) \| \hat{\alpha}(w'|w))$, which is the conditional mutual information $MI(W' : A|W)$. Because this quantity is zero if we have maximal morphological computation, we inverted and normalised in the following way: $1 - MI(W' : A|W)/\log_2 |W|$.

The second concept started with the complementary assumption that the current world state W had no influence on the next world state W', i.e., that the world dynamics kernel is given by $\tilde{\alpha}(w'|a)$. Morphological computation was then quantified as the error from the assumption, given by the weighted Kullback-Leibler divergence $\sum_{w,a} p(w,a) D_{KL}(\alpha(w'|w,a) \| \tilde{\alpha}(w'|a))$, which equals the conditional mutual information $MI(W' : W|A)$.

Both concepts were analysed and quantifications were derived, which didn't require knowledge about the world, but could be calculated from intrinsically available information only. At that time, we could not determine which of the two concepts would capture morphological computation best, although both concepts and their intrinsic adaptations lead to different results in a specific configuration ($\psi = \phi \approx 0$).

Our intention in this publication is to answer this question. For this purpose, we follow a different approach to quantify morphological computation, by starting with the mutual information of $MI(W' : (W, A))$ and decompose it into the shared, unique and synergistic information, as described in the previous section. Replacing X, Y, Z by W', W, A in Eq. (5), we obtain the following decomposition:

$$\begin{aligned} MI(W' : (W,A)) = SI(W' : W; A) + UI(W' : W \setminus A) \\ + UI(W' : A \setminus W) + CI(W' : W; A) \end{aligned} \tag{8}$$

By Eq. (7), our previous concept two, the conditional mutual information $MI(W' : W|A)$, is given by the sum of the unique information $UI(W' : W \setminus A)$ and the synergistic information $CI(W' : W; A)$:

$$MI(W' : W|A) = UI(W' : W \setminus A) + CI(W' : W; A). \tag{9}$$

The examples we have discussed in the introduction (insect wing and Passive Dynamic Walker) suggest to use the unique information $UI(W' : W \setminus A)$ to quantify morphological computation, because it captures the information that the current and next world state W, W' share uniquely. The next section presents numerical results to investigate how the conditional mutual information $MI(W' : W|A)$ and the unique information $UI(W' : W \setminus W)$ compare with respect to quantifying morphological computation.

Experiments

Experiments are conducted on the parameterised model of the sensorimotor loop (see Fig. 1 and Eqs. (1) to (4)). As stated earlier, we set $\tau = 0$, i.e. the world state W is drawn with equal probability ($p(w = -1) = p(w = 1) = 1/2$), and $\zeta \gg 0$ such that the sensor state S is a copy of the world state W. This leaves four parameters for variation, namely the three world dynamics kernel parameters ϕ, ψ, ω and the policy parameter μ. We decided to plot the information theoretic quantities only for $\mu = 0$ (see Figs. 2 and 3), i.e., for the case, in which the action A is chosen independently of the current sensor value S and with equal probability. This allows us to investigate the effect of the action A on the next world state W', without any influence of W on A. We also know from previous experiments (see Zahedi and Ay, 2013), that the conditional mutual information $MI(W' : W|A)$ drops to zero for increasing μ. Thus, by Eq. (9), unique and synergistic information also decrease with increasing μ. If A is deterministically dependent on W, it also follows that the unique information $UI(W' : A \setminus W)$ is zero, because A and W are interchangeable. The only quantity that will be larger than zero is the shared information, which, by definition, is not of interest in the context of this work.

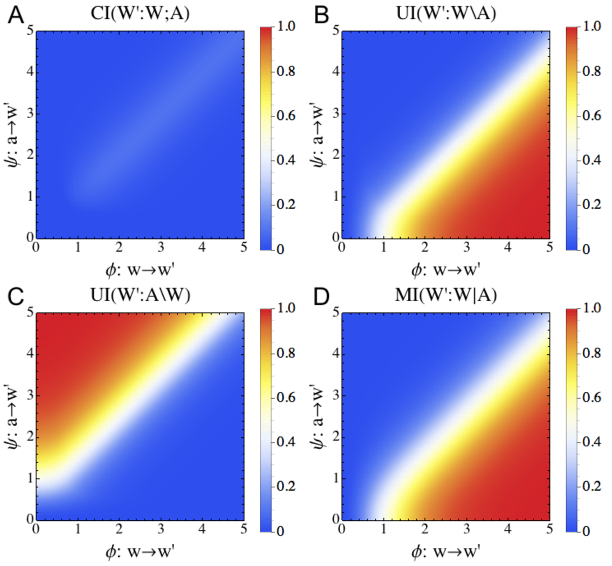


Figure 2: Information decomposition for $\mu = 0.0, \omega = 0.0$

Due to spacial constraints, we decided to plot the information decomposition for varying ϕ (parameter of unique influence of W on W') and ψ (parameter of unique influence of A on W') for two different values of ω (parameter of synergistic influence of W, A on W', see Eq. (1)). This allows us to investigate the effect of the synergistic parameter on the information decomposition. Fig. 2 shows the results for $\omega = 0$, while Fig. 3 shows the results for $\omega = 2$. We will first discuss the results for $\omega = 0$, as they are best comparable with our previous results from (Zahedi and Ay, 2013).

Vanishing synergistic parameter ($\omega = 0$)**:** Fig. 2A shows that synergistic information $CI(W' : W; A)$ is small and only present if $\psi \approx \phi$ (diagonal of the image). This is in agreement with our intuition that ω is the synergistic parameter. The unique information of the action A and the next world state W', denoted by $UI(W' : A \setminus W)$, is shown in Figure 2C. The plot reveals that $UI(W' : A \setminus W)$ is only present when $\psi > \phi$, and it is large whenever ψ is significantly larger than ϕ. Figure 2B shows analogous results for the unique information $UI(W' : W \setminus A)$. In this case, the unique information is negligible whenever $\phi \lesssim \psi$, and

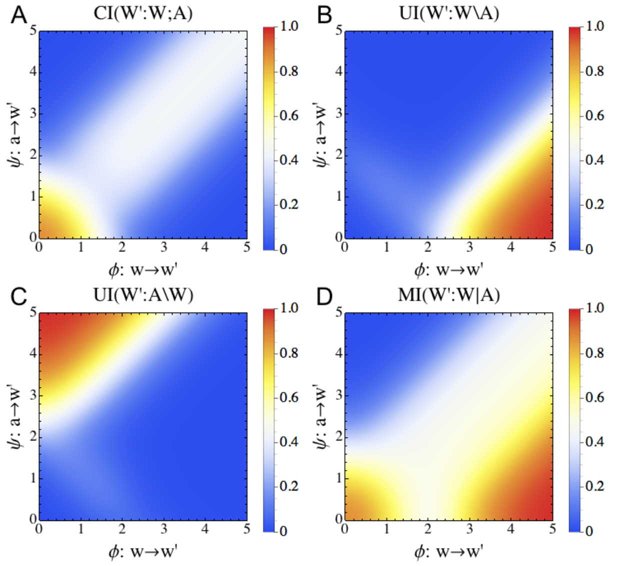


Figure 3: Information decomposition for $\mu = 0.0, \omega = 2.0$

it grows whenever ϕ is significantly larger than ψ. These two plots show that the definition of the unique information, as proposed by Bertschinger et al. (2014), is able to extract the unique influence in a setting in which two random variables actually control, i.e., causally influence a third random variable. Fig. 2D shows the conditional mutual information $MI(W' : W|A)$, which was the second concept of quantifying morphological computation in our previous work (Zahedi and Ay, 2013). As stated earlier, the conditional mutual information is given by the sum of the unique and synergistic information (Eq. (9)). Hence, there is almost no difference between Figure 2B and Figure 2D, except on the diagonal, where the unique information $UI(W' : W \setminus A)$ is slightly smaller.

Positive synergistic parameter ($\omega = 2$)**:** To study the difference between $UI(W' : W \setminus A)$ and $MI(W' : W|A)$, and hence, to compare the new quantification with our former concept, we conducted the same experiments with a value of $\omega = 2$ (see Figs. 3 and 4). Figs. 3A-C demonstrate how the information decomposition can distinguish between the synergistic information and the unique informations, which is exactly what we need to quantify morphological computation. The unique information $UI(W' : W \setminus A)$ captures only the information that the current world state W and the next world state W' share, and therefore, captures the common understanding of morphological computation in the context of embodied artificial intelligence. In the introduction, we presented two examples of morphological computation, which described it as the contribution of the body and environment to a behaviour that cannot be assigned to any neural system or robot controller. The unique information $UI(W' : W \setminus A)$ (see Fig. 3B) captures this notion of morphological computation best, because it vanishes if the synergistic information $CI(W' : W; A)$ (see Fig. 3A) or the unique information $UI(W' : A \setminus W)$ (see Fig. 3C) increases. Given Eq. (9), it is clear that the conditional mutual information $MI(W' : W|A)$ is positive (see Fig. 3D) whenever the unique information $UI(W' : A \setminus W)$ or the synergistic information $CI(W' : W; A)$ is positive. This is problematic for the following reason. Fig. 3D shows a large value of $MI(W' : W|A)$ also for values of $\psi > \phi$, which is counter-intuitive. Furthermore, as Fig. 4 shows (note that the $\phi\psi$ axes are rotated for better visibility), the conditional mutual information is indifferent for a large range of $|\phi - \psi| < d$. Additionally, the conditional mutual information increases for vanishing ϕ and ψ, which again is counter-intuitive, whereas $UI(W' : W \setminus A)$ (see right-hand side of Fig. 4) nicely reflects our intuition. Therefore, we conclude that the unique information $UI(W' : W \setminus A)$ is best suited to quantify morphological computation in the context of embodied artificial intelligence.

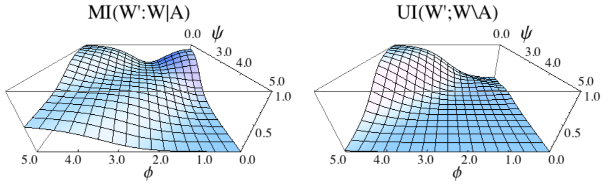


Figure 4: $MI(W' : W|A)$ and $UI(W' : W \setminus A)$ for $\omega = 2$, replotted from Figure 3 to stress the differences between the two measures. The plot on the left-hand side shows more clearly (as compared to Figure 3D) that there is a large domain in which $MI(W' : W|A)$ is indifferent.

Discussion

This work proposes a quantification of morphological computation based on an information decomposition in the sensorimotor loop. In the introduction, morphological computation was described as the contribution of an agent's body and agent's *Umwelt* to its behaviour. Important to note is that both mentioned examples highlighted the contribution of the embodiment that resulted solely from interactions of the body and environment and that cannot be attributed to any type of control by the agent. This is why we propose to use a decomposition of the mutual information $MI(W' : (W, A))$ into shared, unique and synergistic information. This allows us to separate contributions of the embodiment from contributions of the controller (via its actions A) and contributions of both, controller and embodiment.

We showed that the information decomposition is related to our previous work in the following way. The sum of the unique information $UI(W' : W \setminus A)$ and the synergistic information $CI(W' : W; A)$ is equal to the conditional mutual information $MI(W' : W|A)$, which is one of our two earlier concepts for morphological computation. This relation shows the difference of this work compared to our former results. We are now able to quantify exactly how much of the next world state W' is determined by the current world state W, thereby excluding any influence of the action A. Therefore, we propose $UI(W' : W \setminus A)$ as a quantification of morphological computation.

We evaluated the decomposition in a parametrised, binary model of the sensorimotor loop. The world dynamics kernel $\alpha(w'|w, a)$ was parametrised with three parameters, ϕ, ψ, and ω, which roughly relate to the unique information $UI(W' : W \setminus A)$, the unique information $UI(W' : A \setminus W)$, and the synergistic information $CI(W' : W; A)$. For a fixed value of ω, the two parameters ϕ and ϕ were varied to evaluate the information decomposition in the sensorimotor loop. We showed that when the synergistic parameter vanishes ($\omega = 0$), synergistic information is present only for $\phi \approx \psi$. This explains why there is only a marginal difference between $UI(W' : W \setminus A)$ and $MI(W' : W|A)$ in this setting. For a positive synergistic parameter $\omega = 2$, we showed that the synergistic information was positive for a much larger domain, which led to a significant difference between $UI(W' : W \setminus A)$ and $MI(W' : W|A)$. In particular, the condition mutual information $MI(W' : W|A)$ was positive for a larger range of parameter values ψ and ϕ. There is a domain $|\phi - \psi| < d$, for which the conditional mutual information $MI(W' : W|A)$ is positive and indifferent. One would expect to see a higher morphological computation mostly when $\phi > \psi$, despite the fact that synergistic information is present. This shows that $UI(W' : W \setminus A)$ is better suited to quantify morphological computation.

We mentioned in the introduction of this paper that e.g. the flapping of the wing has a component of morphological computation that is independent of any control (the architecture of the wing interacting with the environment) and a component which is induced by the action (flapping of the wings). The synergistic information $CI(W' : W; A)$ captures the second part and should be investigated as a measure for a different type of morphological computation on its own. Unfortunately, this is beyond the scope of this work.

Zahedi and Ay (2013) proposed that a measure of morphological computation could be used as a guiding principle in an open-ended self-organised learning setting. For this purpose, the measure should only depend on information that is intrinsically available to the system. Clearly, this is not the case for $UI(W' : W \setminus A)$. Therefore, future work will include derivations of the information decomposition, which only include intrinsically available information. It would also be interesting to investigate how much a formalisation of the information decomposition can benefit from a consideration of the causal information flow (Ay and Polani, 2008; Ay and Zahedi, 2014). The starting point for our decomposition was the mutual information $MI(W' : (W, A))$, which is a correlational measure and not a measure of causal dependence, as e.g. proposed by Pearl (2000). In currently ongoing work, we are applying the quantification to motion capturing data of real robots.

Acknowledgements

This work was partly funded by Priority Program Autonomous Learning (DFG-SPP 1527) of the German Research Foundation (DFG). The authors thank the reviewers for their valuable feedback.

References

Ay, N. and Polani, D. (2008). Information flows in causal networks. *Advances in Complex Systems*, 11(1):17–41.

Ay, N. and Zahedi, K. (2014). On the causal structure of the sensorimotor loop. In Prokopenko, M., editor, *Guided Self-Organization: Inception*, volume 9 of *Emergence, Complexity and Computation*. Springer.

Bertschinger, N., Rauh, J., Olbrich, E., Jost, J., and Ay, N. (2014). Quantifying unique information. *Entropy*, 16(4):2161–2183.

Clark, A. (1996). *Being There: Putting Brain, Body, and World Together Again.* MIT Press, Cambridge, MA, USA.

Füchslin, R. M., Dzyakanchuk, A., Flumini, D., Hauser, H., Hunt, K. J., Luchsinger, R. H., Reller, B., Scheidegger, S., and Walker, R. (2012). Morphological computation and morphological control: Steps toward a formal theory and applications. *Artificial Life*, 19(1):9–34.

Griffith, V. and Koch, C. (2014). Quantifying synergistic mutual information. In Prokopenko, M., editor, *Guided Self-Organization: Inception*, volume 9 of *Emergence, Complexity and Computation*, pages 159–190. Springer Berlin Heidelberg.

Harder, M., Salge, C., and Polani, D. (2013). Bivariate measure of redundant information. *Phys. Rev. E*, 87:012130.

Hauser, H., Ijspeert, A., Füchslin, R., Pfeifer, R., and Maass, W. (2011). Towards a theoretical foundation for morphological computation with compliant bodies. *Biological Cybernetics*, 105:355–370.

Hauser, H., Sumioka, H., Füchslin, R. M., and Pfeifer, R. (2012). Introduction to the special issue on morphological computation. *Artificial Life*, pages 1–8.

Klyubin, A., Polani, D., and Nehaniv, C. (2004). Organization of the information flow in the perception-action loop of evolved agents. In *Evolvable Hardware, 2004. Proceedings. 2004 NASA/DoD Conference on*, pages 177–180.

McGeer, T. (1990). Passive dynamic walking. *International Journal of Robotic Research*, 9(2):62–82.

McGill, W. (1954). Multivariate information transmission. *Information Theory, Transactions of the IRE Professional Group on*, 4(4):93–111.

Montúfar, G., Ay, N., and Ghazi-Zahedi, K. (2014). A framework for cheap universal approximation in embodied systems. *CoRR (submitted)*, abs/1407.6836.

Pearl, J. (2000). *Causality: Models, Reasoning and Inference*. Cambridge University Press.

Pfeifer, R. (2002). Embodied artificial intelligence - on the role of morphology and materials in the emergence of cognition. In Schubert, S. E., Reusch, B., and Jesse, N., editors, *Informatik bewegt: Informatik 2002 - 32. Jahrestagung der Gesellschaft für Informatik e.v. (GI)*, volume 19, Bonn. GI.

Pfeifer, R., Lungarella, M., and Iida, F. (2007a). Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853):1088–1093.

Pfeifer, R., Lungarella, M., and Iida, F. (2007b). Self-organization, embodiment, and biologically inspired robotics. *Science*, 318(5853):1088–1093.

Pfeifer, R., Packard, N., Bedau, M., and Iida, F., editors (2007c). *Proceedings of the International Conference on Morphological Computation*.

Pfeifer, R. and Scheier, C. (1999). *Understanding intelligence*. MIT Press, Cambridge, MA, USA.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.

von Uexkuell, J. (1957 (1934)). A stroll through the worlds of animals and men. In Schiller, C. H., editor, *Instinctive Behavior*, pages 5–80. International Universities Press, New York.

Williams, P. and Beer, R. (2010). Nonnegative decomposition of multivariate information. *arXiv:1004.2515v1*.

Wootton, R. J. (1992). Functional morphology of insect wings. *Annual Review of Entomology*, 37(1):113–140.

Zahedi, K. and Ay, N. (2013). Quantifying morphological computation. *Entropy*, 15(5):1887–1915.

Zahedi, K., Ay, N., and Der, R. (2010). Higher coordination with less control – a result of information maximization in the sensori-motor loop. *Adaptive Behavior*, 18(3–4):338–355.

Appendix: Computing UI, SI, and CI.

In this appendix we shortly explain how we computed the functions UI and CI. The appendix of (Bertschinger et al., 2014) explains how to parametrize the set Δ_P and how to solve the optimization problems in the definitions of UI, CI, and SI. In our case, where all variables are binary, Δ_P consists of all probability distributions $Q_{\gamma_{-1},\gamma_{+1}}$ with

w'	w	a	$Q_{\gamma_{-1},\gamma_{+1}}(w',w,a)$
-1	-1	-1	$P(w',w,a)+\gamma_{-1}$
-1	-1	+1	$P(w',w,a)-\gamma_{-1}$
-1	+1	-1	$P(w',w,a)-\gamma_{-1}$
-1	+1	+1	$P(w',w,a)+\gamma_{-1}$
+1	-1	-1	$P(w',w,a)+\gamma_{+1}$
+1	-1	+1	$P(w',w,a)-\gamma_{+1}$
+1	+1	-1	$P(w',w,a)-\gamma_{+1}$
+1	+1	+1	$P(w',w,a)+\gamma_{+1}$

The range of the two parameters $\gamma_{\pm 1}$ is restricted in such a way that $Q_{\gamma_{1-},\gamma_{+1}}$ has no negative entries. Since every entry $Q_{\gamma_{-1},\gamma_{+1}}(w',w,a)$ involves only one of the two parameters, Δ_P is a rectangle, bounded by the inequalities

$$\begin{aligned}
\max\{-P(-1,-1,-1),-P(-1,+1,+1)\} &\leq \gamma_{-1},\\
\min\{P(-1,-1,+1),P(-1,+1,-1)\} &\geq \gamma_{-1},\\
\max\{-P(+1,-1,-1),-P(+1,+1,+1)\} &\leq \gamma_{+1},\\
\min\{P(+1,-1,+1),P(+1,+1,-1)\} &\geq \gamma_{+1}.
\end{aligned}$$

To approximately solve the optimization problem we computed the values on a grid and took the optimal value. This simple procedure yields an approximation that is good enough for our purposes.

Quantifying Self-Organizing Behavior of Autonomous Robots

Georg Martius[1,2] and Eckehard Olbrich[2]

[1]IST Austria, Am Campus 1, 3400 Klosterneuburg, Austria
[2]Max Planck Institute for Mathematics in the Sciences, Inselstr. 22, 04103 Leipzig, Germany
georg.martius@ist.ac.at

Introduction In recent years research in autonomous robots has been more and more successful in developing algorithms for generating behavior from a generic task-independent objective. Examples are intrinsic motivations for artificial curiosity, empowerment, homeokinesis, and maximizing predictive information. Independently of its origin, it would be useful to quantify behavior, in order to objectively compare algorithms with each other or even with human or animal movements. We investigate different methods for extracting characteristic measures based on information theoretic quantities.

Example Behaviors We consider behaviors from a hexapod robot with 18 DoF and from a snake robot with 14 DoF when controlled by a simplified predictive information maximizing controller (Der and Martius, 2013). The behavior is generated by an adaptive coupling between sensors and motors in a purely reactive manner – without a central pattern generator nor internal recurrences. In Fig 1 two example behaviors of the snake are shown which we uses here as a brief illustration. The data consists of 10^6 steps of joint position sensors.

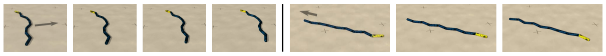


Figure 1: Side-rolling and crawling of the SNAKE. The segments are linked with 2-way hinge joints.

Attractor Dimension and Core Complexity In case of a periodic or quasi-periodic behavior the *attractor dimension* can be considered as a descriptive quantity. For a simple limit cycle the dimension would be one, for a torus it would be two and so on. In addition we want to quantify the complexity of the time series, where the excess entropy (Shaw, 1984) is a natural choice. We study for the first time the resolution-dependence of the excess entropy and propose a decomposition of it into: the attractor dimension, the length-scale of the behavior and the remaining *core complexity*. We find that the excess entropy diverges as $E = c - D\log(\varepsilon)$ for deterministic behavior, where D is the attractor dimension, c is a constant and ε is the resolution (think of a grid size). The constant c contains the overall scale (e.g. amplitudes) of the data. If this is subtracted we obtain the *core complexity* $\hat{c}$ measuring the long term memory of the system: $\hat{c} = c - D\log\varepsilon_{scale}$. The scale can be estimated with different methods, e.g. the variance or more elaborate methods. But how to estimate the excess entropy? We compared estimators based on the continuous mutual information and the correlation integral (Kantz and Schreiber, 2004) with the latter turned out to be more appropriate. Figure 2 shows the excess entropy and respective fits for the behaviors of the SNAKE – both have dimension 1.05 on the coarse (macroscopic) scale ($\varepsilon \in [0.08, 0.8]$). However the constant c would suggest that "side rolling" has a higher complexity (0.75 vs. 0.7), but the new core complexity tells that the "crawling" behavior is more complex. Besides, the "crawling" behavior is 2.5-dimensional on the small (microscopic) scales ($\varepsilon \in [0.0001, 0.01]$).

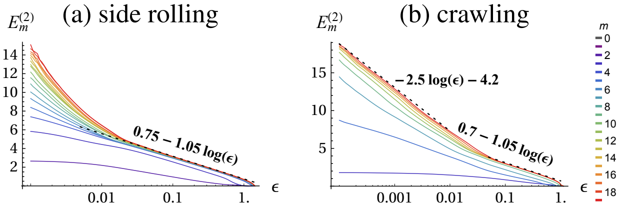


Figure 2: Quantification of two SNAKE behaviors using excess entropy. m is the dimensionality of the phase space reconstruction. Curves for large m must converge and fits yield c and D. Core complexity: (a) $\hat{c} = 2.2$, (b) $\hat{c} = 2.6$ based on rescaling using variance ($\varepsilon_{scale} = 0.25$ and 0.16).

Acknowledgments This work was supported by the DFG (SPP 1527) and the EU (FP7, REA grant no 291734).

References

Der, R. and Martius, G. (2013). Behavior as broken symmetry in embodied self-organizing robots. *ECAL 2013*, MIT Press.

Kantz, H. and Schreiber, T. (2004). *Nonlinear time series analysis*, volume 7. Cambridge University Press.

Shaw, R. (1984). *The dripping faucet as a model chaotic system.* Aerial Press, Santa Cruz.

Modelling Conflict within the Social Networks of Large Multi-Vendor Software Projects using Communicating Stream X-Machines

Richard Alun Williams
Department of Management Science, Lancaster University Management School, LA1 4YX, UK
r.williams4@lancaster.ac.uk

Extended Abstract

There continue to be failures in the management of large IT projects, with anecdotal evidence from the media suggesting that most large public sector projects are not delivered to budget or on time. The fact that we continue to incur failures in managing large IT implementations should come as no surprise however, as they contain a number of areas of risk, ranging from the detailed technical requirements, to the overly ambitious timescales, and the need for large numbers of individual commercial organisations to seamlessly work together in a symbiotic manner. Indeed, some of the larger IT implementations consist of hundreds of customer and third-party team members, who are organised into multiple project teams, and may be located across different geographies and time zones.

With this in mind, it has recently been argued that the increasing size and complexity of these IT projects, leads them to exhibit the behaviours and traits of complex systems (Curlee and Gordon, 2011). Furthermore, with communication and trust issues being cited as the most important aspects of project success (PMI, 2008), it can be hypothesised that an important reason for failure may be the complexity that arises from social interactions within project environments (Hekkala and Urquhart, 2013). As such, we believe that an important reason for project failure may be the emergent behaviours that arise through the social networks within distributed, multiparty project environments; in particular around the areas of communication, commitment, trust, and the resulting conflict that emerges when/if trust breaks down. Conflict in this instance, is broadly defined as an awareness of incompatibilities or perceptions between individuals or groups, which is based on either discrepant views, incompatible wishes or interpersonal relationship issues (Jehn and Mannix, 2001).

Within this study, we have performed semi-structured interviews with project managers involved in large multi–vendor software implementations. This qualitative data has been analysed and transformed into a domain model, which has subsequently been used as the basis for an agent–based model using the concept of communicating stream X–Machines (Balanescu et al., 1999). Through *in silico* experimentation, we are beginning to investigate the propagation of conflict through the social networks in multi-vendor software implementations. In particular, we are investigating the roles that intragroup conflict (between project team members) and intergroup conflict (between different project teams) play in the successful completion of large software implementation projects, involving multiple vendors distributed over multiple geographies and time zones. Our intention is to generate strategies to harness the complexity inherent to large IT projects, and communicate these back to the project management and IT consulting communities so that they may maximise the probability of project success.

Acknowledgements

This work was supported by a SAMS-ESRC-UKCES Management and Business Development Fellowship, grant No. ES/L002612/1.

References

Balanescu, T., Cowling, A., Georgescu, M., Holcombe, M., and Vertan, C. (1999). Communicating stream X-Machines are no more than X-Machines. *Journal of Universal Computer Science*, 5(9):494–507.

Curlee, W. and Gordon, R. (2011). *Complexity Theory and Project Management*. John Wiley & Sons, New Jersey, USA.

Hekkala, R. and Urquhart, C. (2013). Everyday power struggles: Living in an IOIS project. *European Journal of Information Systems*, 22:76–94.

Jehn, K. and Mannix, E. (2001). The dynamic nature of conflict: A longitudinal study of intragroup conflict and group performance. *Academy of Management Journal*, 44(2):238–251.

PMI (2008). *A Guide to the Project Management Body of Knowledge*. Project Management Institute Inc, Pennsylvania, USA, 4th edition.

Dynamic Homeostasis in Packet Switching Networks

Mizuki Oka[1], Hirotake Abe[1] and Takashi Ikegami[2]
[1]Department of Computer Science, University of Tsukuba, Japan,
[2]Graduate School of Arts and Sciences, The University of Tokyo, Japan
mizuki@cs.tsukuba.ac.jp

Abstract

This is an extended abstract of a recent Adaptive Behavior paper (Oka et al., 2014) in which we investigate the homeostatic characteristics of the Internet using a packet switching network (PSN), the fundamental architecture of the Internet. We show that the adaptation introduced in PSN is interpretable as the self-organization of complex itinerant behavior among many quasi-attracting states and provides an example of Ashby's Law of Requisite Variety in action.

A Robust Yet Fragile Nature While the Internet exhibits a great deal of robustness, its fragility, too, has been discussed. Robustness refers to a generic property of stability against perturbation. Fragility, despite common perception, is not an opposite property. For example, in order to take in environmental information, a system should have fragility built into it. Taking in environmental information means lowering entropy of some sort, and doing this requires instability. In fact, the robustness of the Internet is derived from its fragility and it is called Robust Yet Fragile (RYF) characteristics of the Internet (Doyle et al., 2005).

A system would not function if the entire system became unstable due to the incorporation of fragility, thus some type of balancing mechanism such as *homeostasis* is required. How can the RYF property of the Internet can be explained in terms of homeostasis, is the theme of this research. We argue that this is made possible because the dynamical systems of the congestion window size (i.e., the maximum number of packets to be sent per unit of time from each computer), satisfies the Ashby's Law of Requisite Variety (Ashby, 1958). The law states that an active controller requires at least as many states as exist in the controlled system in order to be stable. It also attempts to demonstrate that the effective degrees of freedom of the system change adaptively in response to external perturbations in order to control a system.

To quantitatively study this notion of homeostasis using the Internet, we examine a PSN simulator called ns-2 and discuss its adaptability and robustness. PSNs constitute the fundamental mechanisms of the Internet, which provide adaptive dynamics to the system.

Results and Main Messages The complex dynamics of packet transmission are controlled by the congestion window size (cwnd), whose temporal behaviors change from fixed points to periodic, and finally to chaotic as the number of transmitted packets increases. This transition from a regular to chaotic state corresponds to where the congestion emerges. We further define the throughput as the overall percentage of successfully sent packets in the PSNs to evaluate the RYF nature of PSNs. By externally sending additional packets to perturb the system, we compared the dynamic state of PSNs and the throughput. A dynamical state of a PSN is defined by projecting the cwnd time series onto a reduced-feature vector space using principal component analysis.

As the input ratio increases, the number of dimensions required to reconstruct the original time series increases, leading to an increase in the number of system dimensions, which shows a state transition from a regular to chaotic. When the system's congestion increases, the system begins to generate a large number of states. Beyond a critical input ratio, the cwnd dynamic has no more attracting states. This analysis has been confirmed by applying the perturbation; there is almost no difference in the throughputs between with or without perturbations. The number of dropped packets increases as the input to the system increases due to the congestion, yet the throughput is kept high. This shows the robustness of the PSN system. We claim that this is the realization of Ashby's Law of Requisite Variety in action and RYF is confirmed at the level of PSNs.

Acknowledgements This work was supported by the Japan Society for the Promotion of Science Grant-in-Aid for Young Scientists (#25730184), partially by Grant-in-Aid for Scientific Research on Innovative Areas (#24120704), and partially by Grant-in-Aid for Scientific Research on Innovative Areas (#24650117). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

References

Ashby, R. (1958). Requisite variety and its implications for the control of complex systems. *Cybernetica*, 1(2):83–99.

Doyle, J. C., Alderson, D. L., Li, L., Low, S., Roughan, M., Tanaka, S. S. R., and Willinger, W. (2005). The robust yet fragile nature of the internet. *PNAS*, 102:14497–14502.

Oka, M., Abe, H., and Ikegami, T. (2014). Dynamic homeostasis in packet switching networks. *Adaptive Behavior*, 23(1):50–53.

How social networks shape collective behaviours

Daniel W. Franks[1], A. Jamie Wood[1] and Nikolai W.F. Bode[2]

[1]York Centre for Complex Systems Analysis, The University of York, UK, YO10 5DD
[2] Department of Engineering Mathematics, The University of Bristol, UK
daniel.franks@york.ac.uk

Extended Abstract

Collective motion or swarm behaviour is the synchronized motion of groups of animals such as fish shoals or bird flocks that appear to behave as one body, continually changing shape and direction (Sumpter 2006). The first simulation model of collective motion (Aoki 1982) showed that it could emerge from local interactions between individuals. Several seminal models have since followed this principle (e.g. Reynolds 1987; Couzin et al. 2002). Agent-based models have been important for discerning the simple local rules of individuals that produce emergent group patterns. Such models assume that agents have a sensory range that is limited to a fixed number of nearest individuals or to a perception region of fixed extent. Agents react to the movement of others that are within their sensory range. Interactions often depend on the distance between individuals and can include collision avoidance at short distances, alignment at intermediate distances, and attractive tendencies at long distances.

Social networks exist for many animals and are based on many factors such as preference for familiar individuals or family members. Social networks can be implemented in models to represent the non-spatial topological patterns of preferences of agents. An agent's network connections can then influence their decisions of which other agents to pay more attention to when making a spatial movement decision. However, prior to our research, collective motion models did not consider structured interactions and instead implicitly assume a fully connected egalitarian network (i.e. no discernable preferences). By explicitly capturing social networks in our models, we can study the impact of different structures on the emergent collective behaviour of the swarm.

We have developed a number of agent-based simulations in the biology literature (Bode *et al.,* 2011a, 2011b, 2012a, 2012b) as the first models to **investigate the impact of social networks on collective behaviour** (Figure 1). We will summarise the findings of our research so far, and highlight the possibility for harnessing social networks for improving and manipulating the collective behaviour of swarms. Our models demonstrate that social networks do impact the collective behaviour of groups. We find that groups with more differentiated social structure will navigate to a target quicker and more accurately than groups of individuals in a fully connected egalitarian network, that a sub-group of individuals in central network positions can win conflicts of interests against opposing sub-groups of individuals in less central positions, and that individuals that are more strongly connected in the social network will take a more central spatial position in the group.

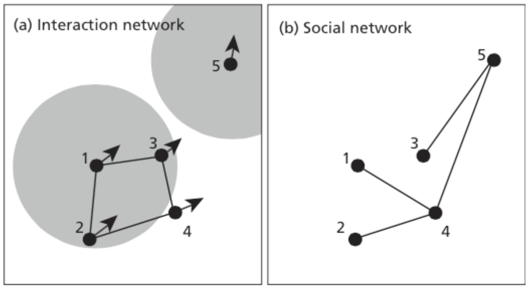


Figure 1: Collective decisions shaped by social preferences. The spatial positions are marked by black circles, with arrows indicating their direction of motion. (a) Illustration of an instantaneous interaction network. The extent of the sensory zones for individuals 1 and 5 are marked by grey regions. Edges in the interaction network are based on which individuals can perceive each other. (b) Illustration of a social network, indicating strong social preferences. Individuals prefer to move in the direction of nearby individuals who they are connected to in the social network. Note how this network contains a connection between individuals 4 and 5, for example, which is not the case in the interaction network. Limited perception can therefore restrict the interaction network to a structure that is different from that of the underlying network of social preferences.

References

Aoki, I. 1982 A simulation study on the schooling mechanism in fish. Bulletin of the Japanese Society for the Science of Fish. 48, 1081–1088.

Bode, N.W.F., Wood, A.J. & Franks, D.W. (2011a) The impact of social networks on animal collective motion, Animal Behaviour, 82(1), 29-38.

Bode, N.W.F., Wood, A.J. & Franks, D.W. (2011b) Social networks and models for collective motion in animals, Behavioural Ecology and Sociobiology, 65(2):117.

Bode, N.W.F., Franks, D.W. & Wood, A.J. (2012a) Leading from the front? Social networks in navigating groups. Behavioural Ecology and Sociobiology, 66: 835-843.

Bode, N.W.F., Wood, A.J. & Franks, D.W. (2012b) Social networks improve leaderless group navigation by facilitating long-distance communication, Current Zoology, 58: 329-341.

Couzin, I.D., Krause J., James R., Ruxton G.D., Franks N.R. (2002) Collective memory and spatial sorting in animal groups. Journal of Theoretical Biology 218:1–11.

Reynolds, C.W. (1987) Flocks, herds and schools: a distributed behavioral model. Computer Graphics 28:25–34.

Sumpter D.J.T. (2006) The principles of collective animal behaviour. Philosophical Transactions of the Royal Society of London B 361:5–22.

At the root of sociality: Working towards emergent, permanent, social affines

Ovi Chris Rouly

Department of Computational Social Science, George Mason University
orouly@gmu.edu

Abstract

Complexity science often uses generative models to study and explain the emergent behavior of humans, human culture, and human patterns of social organization. In spite of this, little is known about how the lowest levels of human social organization came into being. That is, little is known about how the earliest members of our hominini tribe transitioned from being presumably small-groups of ape-like polygamous/promiscuous individuals (beginning perhaps as early as Ardipithecus or Australopithecus after the time of the Pan-Homo split in the late Pliocene to early Pleistocene eras) into family units having stable breeding-bonds, extended families, and clans. What were the causal mechanisms (biological, possibly cognitive, social, and environmental, etc.) that were responsible for the conversion? To confound the issue, it is also possible the conversion process itself was a complex system replete with input sensitivities and path dependencies i.e., a nested complex system. These processes and their distinctive social arrangements may be referred to favorably (as one notable anthropologist has called them) as, “the deep structure of society.” This essay describes applied research that uses discrete event computer modeling techniques in an attempt to model-then-understand a few of the underlying social, environmental, and biological systems present at the root of human sociality; at the root of social complexity.

Introduction

Within the complexity sciences that are the pillars of Artificial Life, computational social science (CSS) often bases its claim of legitimacy on an ability to describe past, ongoing, and future human events through the use of generative computer models that attempt to explain fundamental and emergent human behavior, human culture, and patterns of human social organization (Axtell, 2002; Cioffi-Revilla, 2007). CSS is an applied science: a co-mingled branch of Computer Science and Social Science that pursues its verification and validity from comparisons made between axiomatic, cross-culturally recognized, "first principles" of human behavior. However in spite of this, the literature reports on scant few computer models that test hypotheses on the most basic structure of human social organization. This was the challenge that motivated the current research thread and it led us to consider several related questions.

For example what were the socio-environmental, bio-psychological, and or cognitive drivers that contributed to the initial emergence of the "household?" How did the causal mechanisms of emergent social complexity interact to precipitate such individual and group-level social behaviors as stable breeding-bonds or reciprocal exogamy? Can statistical models (Gavrilets, 2012) which have no socio-temporal interaction memory or explicit socio-spatial context be a reliable indicator of the causes of human sociality? Or, are explanations devoid of biological representation or rich socio-environmental interaction (Kaulakis, R., 2012) but containing seductive oversimplifications of social intercourse based in abstract organizational logic really be plausible explanatory resources? Clearly, innate primate drives like territoriality constrained by pre-adaptive physiological enablers, environmental and social circumscription, and philopatry/dispersal (Parish, 2000) played a role. Moreover, our species seems to have emerged from the milieu of its clade almost in spite of its roots in mixed polygamy over promiscuity without significant benefit of fossil or proxy evidence (Chapais, 2013). So, what realistic set of causal mechanisms (biological, environmental, cognitive, and social) were responsible for our particular species within the hominini tribe to begin its transition from one of likely polygamy-modulated inter-actor promiscuity (sans incest) to one that today purposefully maintains and ultimately exchanges its social affine resources via highly controlled inter-group mating practices? Without at least a few basic answers to these questions the plausibility of models positing their explanatory power over emergent human social organization (and social complexity) might be called into some degree of suspicion. How can we believe the "household" to be the legitimate, primary unit of generative social organization inquiry if the emergence of the household itself cannot be more fully explained? Moreover, is it reasonable to abide the "household" as the basis-unit of computational social science modeling-making if its first principles, its origins, and its fundamental mechanisms are so poorly understood? This is where our research began.

The research described here is work in progress. The purpose in writing this interim report is to create a baseline image of the progress of the work; to establish in public what are our science, our intent, and our tools. The plan is to first identify and then to computationally investigate several of the necessary and sufficient causes of basic human social organization. Together with the current work and its predecessor experiments we have created a single, contiguously coded model whose results emulate emergent, self-aggregating (hominid-like) local-groups that are terrain-situated, mobile, and give spirit to autonomous agent-actors as socially and physiologically plausible as possible. We believe that it is only through such a rich software setting and diverse computational artificial life test bed that one can derive a reliable social science product and a plausible explanatory vehicle for the range of topics that call themselves emergent social complexity.

Methods

We know that the hominini, a tribe that includes the species Homo and its extinct ancestors, e.g., Ardipithecus (Lovejoy, 2009) and Australopithecus, split from the main branch of the African primate phyla roughly 5 – 7 million years ago (mya). And, we also know that Pan, our closest genetic relative in the hominid phylum, last shared an ancestor with us perhaps 7 – 9 mya. Thus, if we can agree that stable breeding-bonds and reciprocal exogamy currently exists in Homo but do not exist in Pan and if we further assume that stable breeding-bonds are a necessary prerequisite to human social organization as it exists today, then we may have a way to logically isolate and confront the confound of this particular nested complex system. As it turns out, it can be shown through phylogenetic decomposition (Chapais, 2009) that the social patterning we refer to as "stable breeding-bonds" in Homo is actually an emergent homoplastic result and not a case of homologous epigenetic inheritance. This will free us in our subsequent analyses to consider only convergent evolution as the determining cause of the processes in question. However, we will still be forced to ask ourselves what manner of circumscription (Carniero, 1988) is in play and how to best go about modeling the remaining complex system. It is upon this layer of theoretic reasoning that we have begun constructing our most recent computer codes.

The following sections describe the construction of a specially prepared individual-based model that it is believed: 1) will offer insight into the causal mechanisms implicit at the lower bounds of social organization theory, and 2) has already shown pertinent, preliminary results. Our goals will be: 1) to test our hypotheses in transportable Java code, and 2) to facilitate the replication of our work by others through the free sharing of that code. The research is purposed to demonstrate emergent, stable, breeding-bonds that may lead toward emergent reciprocal exogamy. The motivating hypothesis is: *Reciprocal exogamy emerged because of innate drives for specific territoriality constrained by evolved pre-adaptive physiological "enablers" consequent to bipedal mobility, social altruism and alliance, environmental and social circumscription, and sexually differentiated philopatry.*

Code Donors

The Java instantiation has a general theme that tends more towards inclusive plausibility than exclusive abstraction. The results enjoy at least the following features: tightly coupled artificial evolution (Darwinian and Baldwinian) expressed through simulated agent cognition and artificial genetics, agent-spatial mobility, 2.5-D simulated terrain with feedback coupled nutrient regrowth carrying capacity, agents with self-adaptive and autonomous learned foraging preferences, agnatic, consanguineal, and uterine kin recognition, mating, disease, malnutrition, infanticide, death by old-age, and single-threaded agent objects bounded by runtime "birth" to "death" encapsulation. In order to abbreviate software development costs, three code donors have been enlisted and severally enlarged from previous works. Features have been added (or removed) to accommodate the specific problems of the current question. Here is an outline of the three code donors used and the names of the respective conference publication titles describing those works. The subsumed donor code features are fully outlined.

A search for the roots of social complexity: Niche adapted agents (Rouly, 2009). This paper and its Java code-base received a poster invitation to the 10th European Conference on Artificial Life held in Budapest, Hungary. This was a spatial agent-based model simulating a niche ecology occupied by fully mobile, sexually dimorphic, and reproductive male/female agents. Each agent had an independent and inheritable artificial chromosome containing eleven genes. The genes were mapped onto graded and expressible biological functions and physiological traits like draught tolerance, temperature sensitivity, robust metabolism, and improved fecundity in small-group settings, as a small example.

1. **Figure 1** depicts a typical social network that often emerged among the agents in the model. What is visualized is a population-inclusive effect called Genetic Drift (Sewell-Wright Effect, Wright, 1932). The image uses a social network to visually articulate the drift that emerged within the chromosomes of the agent population over 4,259 years-of-days.
2. *A prototype, multi-agent system for the study of the Peopling of the Western Hemisphere* (Rouly and Crooks, 2010). This applied work extended the previous research code base. The extensions included highly-detailed, agent-terrain spatial interactions, better agent foraging autonomy, and improved socio-spatial mobility. This work was delivered to the 3rd World Congress on Social Simulation in Kassel, Germany. Specific requirements for empirically-derived environment and terrain components built for this Java model demanded that enhancements be made to the code base so as to more fully articulate daily climate, flora, fauna, and water resource updates. The new features included daily and seasonal updates of the ecology through multi-threaded execution that took better advantage of multi-core, multi-processor technologies.
3. *Sexually differentiated philopatry and dispersal: A demonstration of the Baldwin effect and genetic drift* (Rouly and Kennedy, 2011). This work further extended the code base. Here we refactored previous work in order to introduce two new features. The first feature was the addition of an empirically grounded process of sexually differentiated philopatry and spatial dispersal. The behavior of spatial dispersal occurs in many primates when they become sexually mature. The second feature involved a new agent behavioral control mechanism that simulated a cognitive (adaptive) process in addition to the existing innate (biogenic-reactive) processes. This new work demonstrated self-aggregating cohorts (fission and fusion) and increasingly plausible, situated and embodied, hominid-like agent behavior. Finally, the agents in this new experiment were required to make individual foraging choices that de-conflicted their

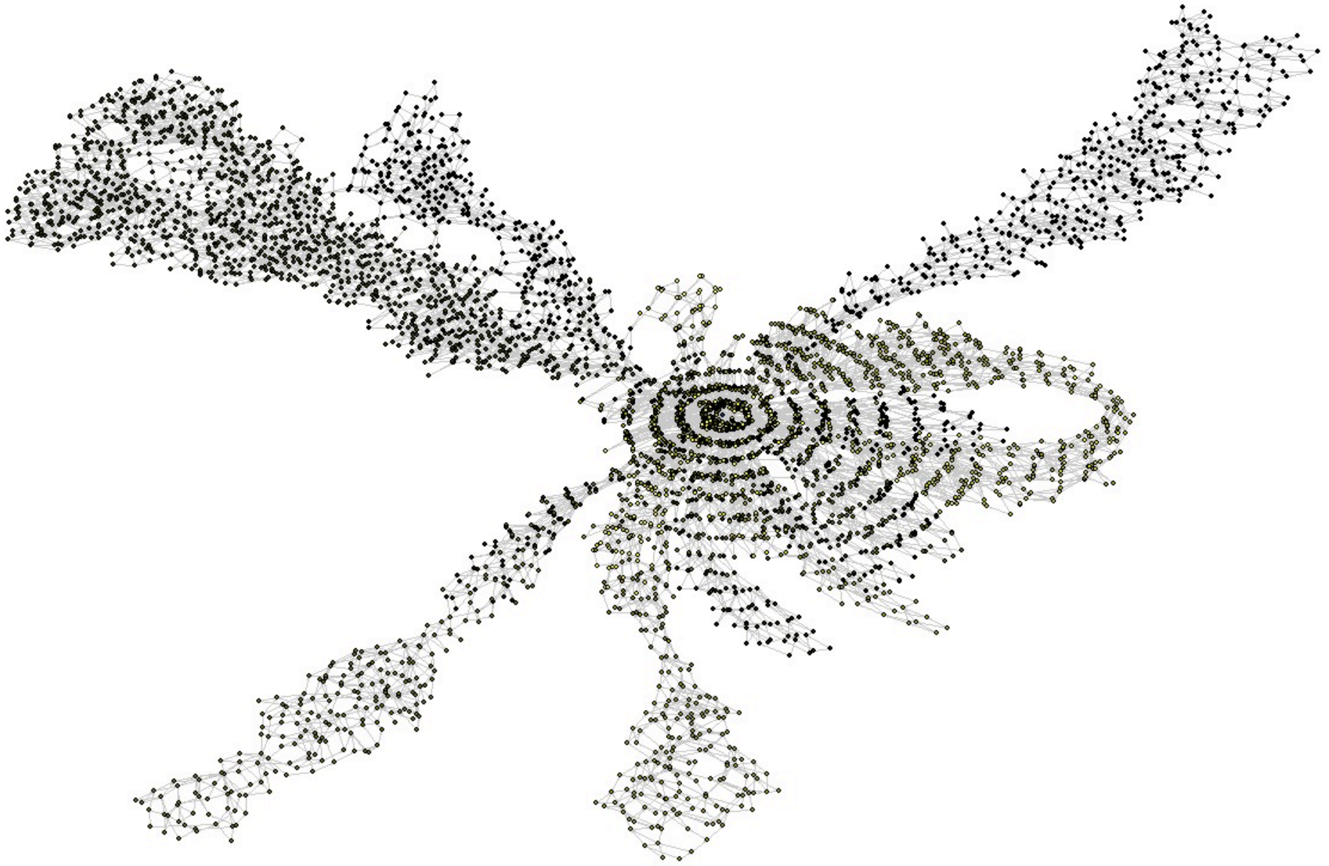


Figure 1 This illustration captures the emergence of the Sewell-Wright Effect within the genetics of an artificial agent social network. The temporal-spatial progress of the emergent process appears as a pattern of expanding “tree rings” among the interconnected network nodes. The radial arms are spatially separated cohorts of promiscuously “mating” agents. The “evolution” of genetic content is shown as a subtle color shift from the center “rings” (first generations) to the outer “rings” (last generations). Bridging connections between radial arms are the “mating” activities that occurred between otherwise spatially separated agent groups, or cohorts, as time passed and the cohorts became more and more spatially distant. Longer radial arms represent more adaptive genetic results in a particular part of the niche ecology.

mother/infant taught eating practices with terrain food-choice availability in real-time. By isolating a few independent variables in what had by now become a rich and stable software test bed, agent perceptual conflicts arising during foraging allowed us to observe the results of simulated cognitive dissonance. As the process of dissonance resolution executed within each individual agent an emergent, quantifiable, species-changing genetic result was observed but in abbreviated evolutionary timescales. **Figure 2** revisits the emergent genetic drift produced by the previous baseline code but in the context of this new experiment. Here we see a quantifiable result: the skewed distribution of genetic values after Baldwinian evolution.

The Current Code

Logical extensions made for the current research have required new code to be written. Consequently, the donor code has been significantly refactored and nominally three new Java classes named TrueRNG, Socioecology, and Groups, were created. Additionally, genes for social altruism and alliance were added for independent variable comparison testing. Great care was taken not to introduce, or tolerate, regression errors in the code donor base-classes.

The software can now dynamically accommodate maps of any size so long as the incoming graphic is based on square kilometer increments. As the map is read-in, any number of colors appearing on the map can be tagged and used as "land features" within the model. This can and does include 2.5-D relief features. Once incoming "land features" are

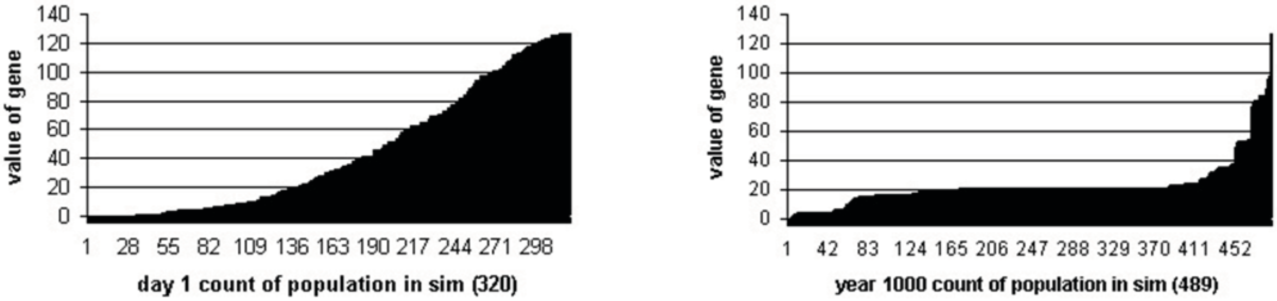


Figure 2 On the left is an illustration of the initial distribution of a representative gene value in the starting population. The number of agents was 320 on day 1 and the values were initialized as a Gaussian distribution. On the right is the skewed result of the same gene in the surviving population (many generations later) in year 1000. The distribution is clearly skewed to the right. The population had grown to 489 agent members.

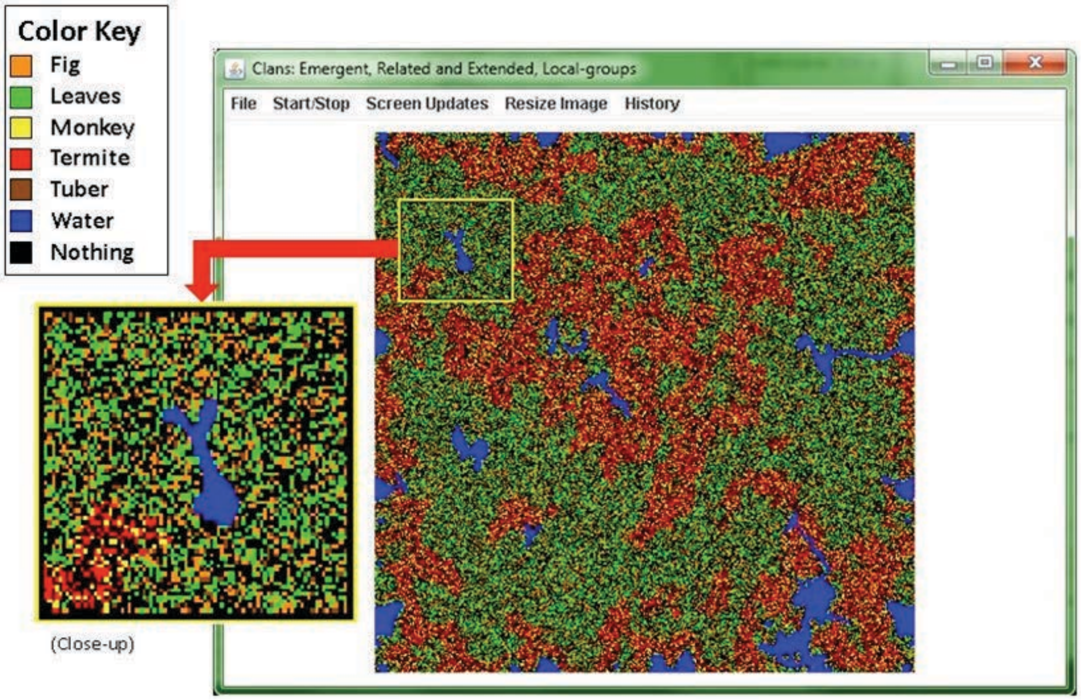


Figure 3 This is a screen capture of the current model based on a 4km X 4km 2.5-D terrain map, or habitat. The key to the left identifies the value of each of the pixels in the map. Each pixel – and the behavior of the hominid agents in the simulation – is based on the assumption that a pixel is a 10m X 10m area. The forageables Fig, Leaves, Monkey, etc., are grouped by 2.5-D relief height. The color of each pixel represents and provides an agent-harvestable calorie or water volume, respectively. The "greenish" areas are lowlands and the "reddish" areas are highlands. Water, in "blue," is the lowest terrain height.

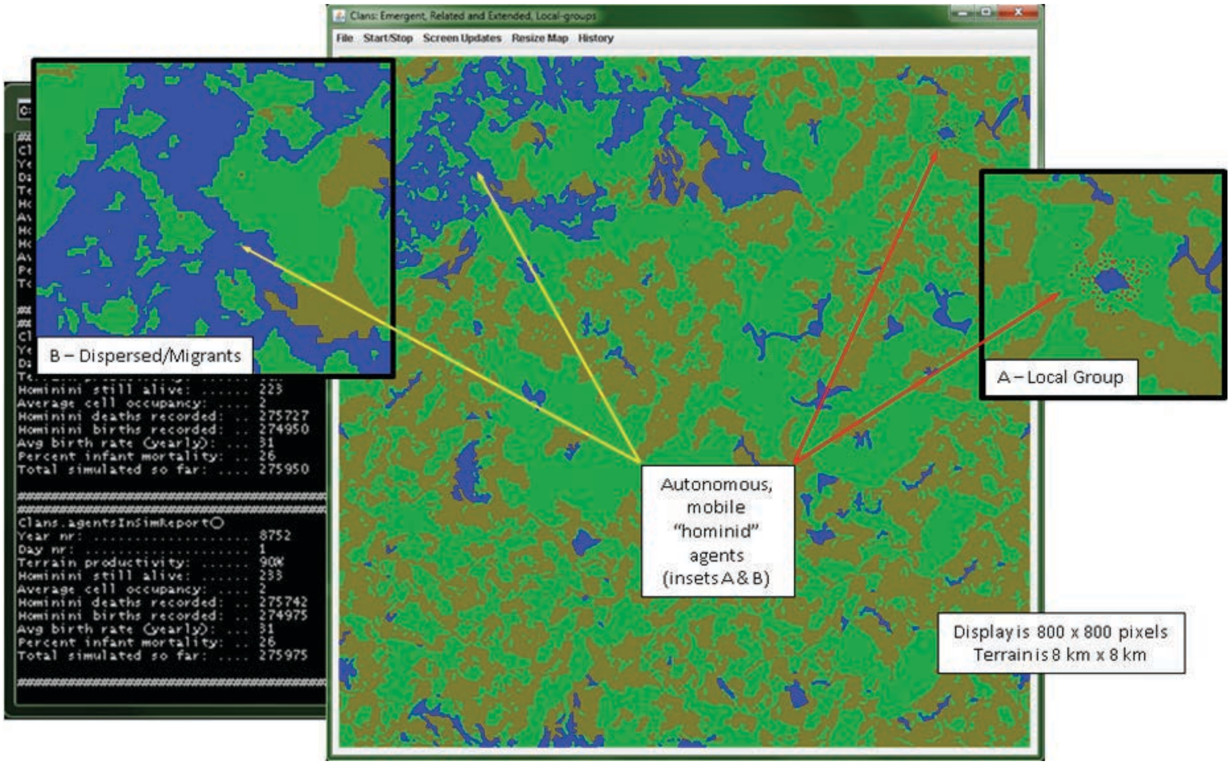
recognized, the distribution of a plausible set of calorie providing forageables can be placed on the terrain. **Figure 3** shows one such distribution of forageables. During the execution of the simulation "land features" like climate, flora, fauna, and water resources, etc., are updated (per epoch) in separately executing threads per each square kilometer. This is aimed directly at improving bandwidth utilization in multi-core processors with large RAM capacity. Together these changes allow topographical maps of spatial environments past, present, or abstract to be imported for study. Finally, because of the refactoring, the Ecology related classes of Climate, Flora, Fauna, Terrain, and now Groups, each are in contact with the agent cohorts during multi-threaded execution. This was an important achievement adding plausibility and efficiency by way of quasi-concurrency. The Groups code provides external modules access to statistical and accessor methods addressing the terrain cohort array lists. Now that the Ecology classes are multi-threaded their combined daily (one epoch) maintenance loops take less real-time to execute. So much was the improvement that forageables can now be dynamically (feed-back) controlled by hominid foraging. It is entirely possible for a hungry horde of hominids to wipe out an entire terrain cell's productive capacity or to drain one of the randomly placed small water basins on a daily basis. Or, to the opposite, allow a terrain cell to recover if sufficient time has passed and the cell forageables are unharvested. This feature is new to the research thread but was inspired by similar work done in the Sugarscape series (Epstein and Axtell, 1996).

An entirely new addition to the work is the use of a diode-noise-based TrueRNG® random number generator. This is a hardware device (a USB dongle). It has shown itself, in empirical tests, to be able to typically produce no more than one or two integer repetitions in a 64-bit sequence of over 1 billion uniform random number generation attempts. While this is far from perfect, it is several orders of magnitude better than the factory Java class running the same test. In the research described here the device is used as an entropy source and a generator of random number seeds for the hominid agents once a year on their individual birthdays. The result of adding this TrueRNG® to the research has been to "flatten" the stochastic and often sudden excursions associable with population crash and or explosion. The "downside" is that each run is an entirely unique random/stochastic proof of the validity of the system model but is, by itself, only quantifiable by stochastic repetition.

The Socioecology class is entirely new and supports many significant and novel inter-hominid socializing activities and in-group/out-group recognition results. A challenge of this research is that the experimental definition strictly allows us to only "precipitate" the emergence of stable breeding-bonds and the follow-on occurrence of agnatic memory and exchange-capable affine relations given an initially promiscuous hominid agent base – but not directly cause the related behaviors of polygamous organization or monogamous pair-bonding. In fact, there is explicit negative value attached to hard-coding any of the normal inter-social identifiers and or social behaviors associable with the "household."

Results

We report here intermediate (in-progress) results. **Figure 4** is a screen capture taken from one model (simulation) run.


Clans: Emergent, Related and Extended, Local-groups
File
Start/Stop
Screen Updates
Resize Map
History
B – Dispersed/Migrants
A – Local Group
Autonomous, mobile "hominid" agents (insets A & B)
Display is 800 x 800 pixels
Terrain is 8 km x 8 km

```
Hominini still alive: ...... 223
Average cell occupancy: .... 2
Hominini deaths recorded: .. 275727
Hominini births recorded: .. 274950
Avg birth rate (yearly): ... 31
Percent infant mortality: .. 26
Total simulated so far: .... 275950

Clans.agentsInSimReport()
Year nr: ................... 8752
Day nr: .................... 1
Terrain productivity: ...... 90%
Hominini still alive: ...... 233
Average cell occupancy: .... 2
Hominini deaths recorded: .. 275742
Hominini births recorded: .. 274975
Avg birth rate (yearly): ... 31
Percent infant mortality: .. 26
Total simulated so far: .... 275975
```

Figure 4 The console display here is showing the results for the simulated year 8,752 after program start. The total count of agents simulated over hundreds of generations was 275,975 with 275,742 deaths recorded. When this screen was captured 233 agents remained. The red pixels near the bodies of large standing water (in the insets) are promiscuous hominid agents.

The simulation epoch is one day and a simulated year is 364 days. In the screen capture, the simulation has been running for 8,752 years-of-days (or 3,185,729 epochs). In the console window to the left of the terrain image there is scrolling output. The output displays a series of cumulative sums taken each year over all previous years. On the map the hominids are visible as “red” pixels. The main local group is near the perimeter (shoreline) of a large stationary body of (blue) water in the upper-right corner of the habitat. This habitat is 800 X 800 pixels square. Each pixel along with the agent behaviors are scaled to 10m X 10m. Thus, the entire habitat is a torus grid 8km X 8km in size and represents 64 simulated square kilometers.

At program start in the run shown in the screen capture, there were 1,000 groups of 1 agent seeded onto the habitat terrain. Placement of those seed-groups (referred to as cohorts) was random. The seed population was composed of mixed sex individuals ranging in age from 6 to 35 years. The majority of the agents were between 15 and 35 years of age. None of the females were pregnant or nursing at program start. In the run shown and within a few generations, seed group numbers had dropped to several hundred and many “small” local-groups (cohorts) had coalesced around many widely scattered bodies of water. Within a few hundred years, the cohort in the upper right corner of the Figure emerged as the only local-group remaining. After over 8,000 years-of-days, agent spatial choices had clearly become the stochastic product of autonomous decision making paired with the spontaneous interactions of breeding members within their forageable (niche) environments.

During this particular simulation (or run) the population of simulated hominids ranged in number from a low of near 100 individuals to a high of a few hundred (after initialization). Sadly, the entire colony in this particular run expired just over one hundred years (8,863 actual date) after this image was taken. Preliminary post-mortem analysis suggests the collapse was associated with events surrounding momentary colony membership spatial dispersal, a random shift in new offspring sex-ratios, and a seasonal swing in environmental carrying capacity.

As stated earlier, this is an intermediate report containing progress and observations of model performance. Yet, as described in the preceding paragraph, the experiment has already delivered relevant patterns of interaction between our base-line, promiscuous hominid-like agents, their environment, and their breeding groups. While we are not presenting final quantitative or qualitative conclusions here, regarding the motivating hypothesis, our accumulating social networks, genetic analyses, and socio-environmental results are suggesting that perhaps with the addition of genetics-based social altruism and alliance alone we have begun to see promiscuity changing into polygamy and possibly semi-permanent breeding-bonds. That said, our data is simple and is comprised of longitudinal and statistical results. Consider: Every epoch (day) recently deceased members of the population are expunged from the simulation. When the agents are removed a complete “death certificate” is made for each individual. That “death certificate” contains running data like agent ID, date and location of simulated birth, death, age at weaning, biological mother and father IDs, agent actual weight, caloric and water requirements, preferred prey, cause of death, and several more data points including a complete sample of the genetic material of the agent. Although it is currently only used for diagnostic testing and software development verification, a complete listing of each of the siblings belonging to the agent and the birth order of those siblings, assuming there were any, can be “listed.” Along with this, a listing of all cohort acquaintances and their respective daily interaction accounts can also be produced.

Discussion

While the overall research thread is extensive, it profits greatly from the described, incremental improvements within and over its predecessor donor codes. It relies heavily on those donor codes described in the Methods section. However, given space constraints, more about the donor codes cannot be discussed. The important new Socioecology class needs our final attention.

Introduced in the Methods section, Socioecology is new code to this research thread. It endows each individual hominid with a dynamic, random access, “social group-memory” capability and several logical operators over that memory. It operates such that as a hominid agent encounters others the self can compare the identity of self and the identities of the others, storing the IDs of the others, and thus later having the capacity to recognize the other agents throughout the lifetime of self. Moreover, each day (epoch) self and other occupy the same spatial cohort (a single 10m X 10m terrain cell) a discrete counter is incremented for the specific other. That counter is available to all logical social operators in the class. Additionally, three important new social functions in the class were created; they are uterine kin recognition, Westermarck recognition, and consanguineal kin recognition.

Uterine Kin

Building on these new inter-agent checks, the Socioecology class has code supporting explicit knowledge of matrilineal family constellations. For example, all primates appear to be able to differentiate some of their dyadic behaviors based on uterine kinship. Thus, the new code can directly store and manipulate others that include siblings, mother, children (if self is female), and logical relations like birth-order. For example, dominance hierarchies are often associated with birth-order (Sapolsky, 2005). Methods also exist to permit agnatic and consanguineal kin identification when those instances occur. The time invested in creating these new kinship identification methods has already paid dividends in the context of incest avoidance (Rodseth, 1991). Previously, interpersonal methods inherited from the donor code-base could only remember one other/father/son/daughter relationship at a time. Thus, while first generation incest was forbidden by rule (and it was very effective) it was also possible within a sufficiently large mating group that incest could occur over a longer period of time between consanguineal kin. Although consanguineal incest is theoretically still allowed to happen, at least when it happens now, there is a console message sent to the operator and a statistic taken for the event. Finally, however, because every hominid knows its mother and its siblings and the mother knows all of her children, incest avoidance based on uterine kin recognition (a necessary component of any society having patrilocal residence like Pan) is controlled in the new code.

Westermarck Recognition

The code also contains a Westermarck (1921) function based on the hypothesis that "familiarity" may implicitly contribute to incest avoidance. This heuristic suggests that primate sexual relations are forbidden based (effectively) on the number of days self has been in contact with other. In the Socioecology code this is calculated as an average number of days self has been in the proximity of other and a threshold value comparison taken. The Boolean function fires if the number of contact days with other meets a nominal threshold for avoidance. Together with uterine kin recognition and rejection, Westermarck keeps incest rates well below 0.1% over several generations of mating occurrences without benefit of any other explicit incest avoidance rules.

Consanguineal Kin

In promiscuous breeding troops (like those of Pan), any sufficiently old biological father of any troop offspring may not be cognitively certain of his own paternity in the context of any living constellation of infants, juveniles, or younger adults. This appears also true, by reflection, from the viewpoint of younger male and female troop members onto any older extant male. However, the younger members may, by virtue of their individual memory of continuous social interactions, i.e., by virtue of the Westermarck function, be able to make informed inferences regarding the bio-social relationship between self and the older other. If this hypothesis continues proving itself valid in testing, it will facilitate several simple coding mechanisms for generating bilaterally emergent agnatic relationship recognition. This is a primary goal of the research thread and a latent function within the Socioecology class.

The research has only in the last few months begun to produce results. From the Code Donors we inherited a base of situated, stable, sexually dimorphic male and female agents. These were agents that had previously enjoyed self-directed foraging and sexually dimorphic philopatric/dispersive behaviors, cognitive features, artificial genetics and biology, and all of the rest. To them we added social altruism and alliance, and an increased capacity for highly-social interaction. Given the foregoing, we look forward to soon being able to report emergent patterns of abstract, implicit kinship and social networks showing a history of permanent breeding-pairs within the populations that inhabit the model.

Summary

This article has disclosed an ongoing computer-based experiment. The experiment uses a computer simulation technique called agent-based modeling (ABM) as the basis of its work. For the last few years this research has led to the identification and accumulation of a set of self-organizing social properties, hominid-inspired behaviors, pristine environments, and physiological enablers believed to exist at the least-organized end of every complex (human) social system. This is the specialized domain of the work.

In particular, the work deals with highly-social populations of explicit, initially promiscuous, primate-like software agents inhabiting 2.5-D virtual environments. We have seen our historical experiments creating plausible, artificial, and vibrant social fabrics within and between situated agents who themselves autonomously demonstrated survival-related and innate small-group social behaviors. In the previous works named in the Code Donors part of the Methods section, we gave testimony and reference to peer-reviewed evidence that our evolving code base has developed agent populations generating spontaneous and emergent social behaviors ranging from community fission and fusion, to voluntary migrations, simulated sexual reproduction, new agent birthing, aging, and death, and now (most recently) what may be semi-permanent breeding-bonds that resemble emergent polygamy from within a wholly promiscuous population.

It is believed that what is contributed by the work reported here is important for two reasons. First, the system model that we adopted for instantiation is inherently detailed and expansive: much as is the subject under study, i.e., natural-life. Metaphorically: This is not research that attempts to sneak a slice or a bit of pie but rather it is an attempt to create a whole pie. This is generative Computational Social Science. It builds on the genre of models instantiated by Epstein and Axtell (1997), Kohler, et. al, (2000), Axtell, et. al (2002), and others and it is inherently deductive in its approach. This work bases its algorithms and conclusions on computations drawn from empirical or empirically derived parameters, objectively substantive relationships, and observable processes (Epstein, 1999). In recent years we have developed better computers and more advanced software engineering techniques. So now the question must be asked, should we not be building and studying more models of similarly complicated and broad-ranging natural-life systems? And, second but more importantly, the simulation about which we report here explores the roots of our own complex, human social structure; at its least-organized end. This is a subject that is known today only by speculation and religion. That alone should be challenge enough for us to harness our technologies and make every attempt to better understand the dynamics of emergent, small-group social behavior.

Our goal was the discovery of new factors contributing to the socio-environmental, bio-psychological, cognitive, and singularly social development of our species. It may be that it is only through explicit simulations, like those disclosed here, that we can visualize the emergence of the structures most fundamental to complex human social organization. Simulations like these allow us to ask “what if” questions; questions that are otherwise unethical, impractical, too expensive, and too time consuming to be tractable by any other means. And, this is of course not an exclusive list.

Our task was to attempt to bring about emergent and permanent agent breeding-bonds, breeding-pairs, family units, clan-like social structures, or nascent reciprocal exogamy within an otherwise detailed, wholly promiscuous (primate-like) population. By step-wise iteration we have found that it appears that if social altruism and alliance (expressed as the voluntary sharing of food resources coupled with post-benefit preferential relocation) were sexually differentiated traits available to every member of a test population then, we may have taken the first steps towards our goal. Said more simply, by adding a single independent variable (adding gene-based traits for altruism and alliance) we have moved an explicit, promiscuous population incrementally toward polygamy, semi-permanent breeding pairs, and or both. Clearly, this is an experiment that tends more towards inclusive plausibility than exclusive abstraction and lingering doubt. Primates, after all, are very complex social beings.

Conclusion

As is the case with many complex systems models, model initialization can be difficult due to input parameter sensitivity. This model is no different. It has been said that complex systems models experience a “settling period” when they first begin to run (personal conversation with R. Axtell in Fairfax, Virginia, 2015). This occurs as agent schema and parameters are filled with actual runtime versus initialized values. As was noted previously, the use of the TrueRNG® dongle does appear to help smooth population growth dynamics in this latter regard and this is a good thing. But, to complicate matters, we also have implicit constraints associable with our hominid prototype that demand our populations have and keep membership numbers small relative to any habitat size under study.

Considering all of the foregoing and conditions associable with minimized genetic diversity, issues known to plague the prototype species have become issues within our model too. For example, our agents can suffer from problems associated with localized over-grazing if their habitat is too small or their numbers grow to large before troop fission. That said, in an extension to this work we may yet add territorial patrols (Mitani, 1979) to the behavioral ecology of the simulated hominid population (Wrangham, 1975). Of course, we will need a larger artificial habitat. But, in the near term, we can report that with only the addition of genetics-based altruism and social alliance, we have already seen incremental progress towards our goal of emergent, permanent, social affines.

References

Axtell, R. L., Epstein, J. M., Dean, J. S., Gumerman, G. J., Swedlund, A. C., Harburger, J., & Parker, M. (2002). "Population growth and collapse in a multiagent model of the Kayenta Anasazi in Long House Valley." *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 3), 7275-7279.

Carneiro, R. L. (1988). The circumscription theory. *American Behavioral Scientist*, 31(4), 497-511.

Chapais, B. (2009). *Primeval kinship: How pair-bonding gave birth to human society*. Harvard University Press.

Chapais, B. (2010). The deep structure of human society: primate origins and evolution. In *Mind the gap* (pp. 19-51). Springer Berlin Heidelberg.

Chapais, B. (2013). Monogamy, strongly bonded groups, and the evolution of human social structure. *Evolutionary Anthropology: Issues, News, and Reviews*, 22(2), 52-65.

Cioffi-Revilla, C., Luke, S., Parker, D. C., Rogers, J. D., Fitzhugh, W. W., Honeychurch, W., Fröhlich, B., De Priest, P., & Amartuvshin, C. (2007, January). Agent-based modeling simulation of social adaptation and long-term change in Inner Asia. In *Advancing Social Simulation: The First World Congres*s (pp. 189-200). Springer Japan.

Epstein, J. (1999). Agent ‐ based computational models and generative social science. *Complexity*, 4(5), 41-60. doi:10.1002/(sici)1099-0526(199905/06)4:5<41::aid-cplx9>3.3.co;2-6

Epstein, J., & Axtell, R. (1996). *Growing artificial societies: social science from the bottom up*. Brookings Institution Press.

Epstein, J. M., & Axtell, R. (1997). Artificial societies and generative social science. *Artificial Life and Robotics*, 1(1), 33-34.

Gavrilets, S. (2012). Human origins and the transition from promiscuity to pair-bonding. *Proceedings of the National Academy of Sciences*, 109(25), 9923-9928.

Kaulakis, R., Zhao, C., Morgan, J. H., Hiam, J. W., Sanford, J. P., & Ritter, F. E. (2012). Defining factors of interest for large-scale socio-cognitive simulations. In *Proceedings of ICCM-2012-Eleventh International Conference on Cognitive Modeling* (pp. 117-122).

Kohler, T. A., Kresl, J., Van Wes, Q., Carr, E., Wilshusen, R. H.: Be There Then: A Modeling Approach to Settlement Determinants and Spatial Efficiency Among Late Ancestral Pueblo Populations of the Mesa Verde Region, U.S. Southwest. In Kohler, T. A., Gumerman, G. J. (eds.): *Dynamics in Human and Primate Societies: Agent-Based Modeling of Social and Spatial Processes*. Oxford University Press, Oxford UK (2000) 145–178.

Lovejoy, C. O., Suwa, G., Simpson, S. W., Matternes, J. H., & White, T. D. (2009). The great divides: Ardipithecus ramidus reveals the postcrania of our last common ancestors with African apes. *Science*, 326(5949), 73-106.

Mitani, J. C., & Rodman, P. S. (1979). Territoriality: the relation of ranging pattern and home range size to defendability, with an analysis of territoriality among primate species. *Behavioral Ecology and Sociobiology*, 5(3), 241-251.

Parish, A. R., de Waal, F., & Haig, D. (2000). The other "closest living relative": How bonobos (Pan paniscus) challenge traditional assumptions about females, dominance, intra-and intersexual interactions, and hominid evolution. *Annals of the New York Academy of Sciences*, 907(1), 97-113.

Rodseth, L., Wrangham, R. W., Harrigan, A. M., Smuts, B. B., Dare, R., Fox, R., King, B., & Wolpoff, M. H. (1991). The human community as a primate society [and comments]. *Current Anthropology*, 221-254.

Rouly, O. C. & Crooks, A. (2010). A prototype, multi-agent system for the study of the Peopling of the Western Hemisphere. *3rd World Congress on Social Simulation*, 2010. Kassel, Deutschland. 6th-9th September 2010.

Rouly, O. C. & Kennedy, W. G. (2011). Sexually differentiated philopatry and dispersal: A demonstration of the Baldwin effect and genetic drift. *Computational Social Science Society of America Annual Conference, 2011*. Santa Fe, New Mexico, USA. October 9-12, 2011.

Rouly, O. C. (2009). *In Search of the Roots of Social Complexity*. George Mason University. Unpublished manuscript.

Sapolsky, R. M. (2005). The influence of social hierarchy on primate health. *Science*, 308(5722), 648-652.

Thompson, M. E. (2013). Reproductive ecology of female chimpanzees. *American journal of primatology*, 75(3), 222-237.

Westermarck, E. (1921). *The history of human marriage* (Vol. 2). Macmillan.

Wrangham, R. W. (1975). *Behavioural ecology of chimpanzees in Gombe National Park, Tanzania*. Ph. D. thesis, Univ. of Cambridge.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the 6th International Congress on Genetics*, 1, pp. 356–366.

The Circular Logic of Gaia: Fragility and Fallacies, Regulation and Proofs

Inman Harvey

Evolutionary and Adaptive Systems Group, University of Sussex, Brighton, UK
inmanh@gmail.com

Abstract

Gaia Theory makes controversial systems-level claims about how environmental factors on a planet tend to be regulated towards conditions favourable to biota. Daisyworld models show how such phenomena may arise, but their subtleties are often misunderstood. Here the core concept of Gaian regulation is shown in an ultra-minimalist model, and defined in terms of fragile hysteresis loops. Insights thus gained explain many fallacies and misunderstandings held by critics and advocates alike. Gaia Theory and Darwinian evolution are shown to be complementary not antagonistic. General results are proved for the inevitability of Gaian regulation at steady state in systems of arbitrary size and complexity under very broad conditions.

Introduction

Gaia Theory makes controversial claims about system-level properties of interactions between biota and environment in a bounded world. In some sense, it is claimed, environmental factors tend to be regulated favourably to the biota – but in what sense, and why? What is being regulated? Is something (what?) being optimised? Is it compatible with evolution? Does it depend on a lucky (or deliberate) choice of bio-environmental interactions? We build on Daisyworld (DW: Lovelock, 1983; Watson and Lovelock, 1983), a simple Artificial Life model, to answer these questions. The unfamiliar Gaian circular logic is easy to misinterpret, by critics and advocates alike; confusion has bred mistrust on both sides, not least on the relationship between Gaia Theory and evolution. We aim to clarify how such confusion arose.

The strategy of this paper is to first present the core of this circular logic (Fig. 1) in the context of the simplest model possible: a 2-variable, 1-parameter toy, abstract mathematical model. Gaian regulation (GR), defined in terms of Viability-Zones in Perturbation Space, will be demonstrated. This corresponds to a specific form of hysteresis loop, a zone of 'fragile' bistability. The circular causation $B \rightleftarrows E$ upsets many prior expectations, and is used to illustrate common fallacies and misunderstandings. Thus it may act as an intuition pump when considering the real world of biology and environment, or synthetic worlds of man-made systems.

The second part of the paper generalises from this 2-variable model to *n*-variables and interactions of any complexity, subject only to very broad conditions. GR, thus defined, extends to arbitrary systems of any size: this is 'inevitable Gaian regulation', perturbations are automatically countered so as to widen, not lessen, the viability range.

It is shown that this viability-based GR is indeed in accord with the core of the original DW of W&L, though even there it is necessary to distinguish the core from added complexity that is not essential to GR. Clarification of the core concept both makes possible the very general theorem proved here and indicates where translation to the real world may be fruitful. The results pertain to equilibria and not to transient responses.

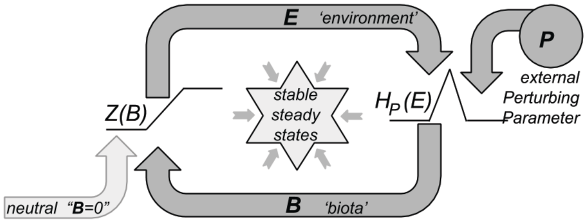


Figure 1. Circular causation: perturbations P regulate the stable steady states of the $B \rightleftarrows E$ circuit. B is viable (>0) for some range of P values. It will be proven that if the circuit is broken by a hypothetical neutral 'switch' pictured at left, B is viable for a decreased range of P (or at best, the same range). I.e. the unbroken circuit typically increases B-viability.

Principles and Anecdotes. Most studies of DW (Wood, 2008) make extensions to the core so as to show new phenomena or to fit the model better to real data. This paper has the opposite motivation: in dismissing any such extensions as 'merely anecdotal' insofar as they are not generalisable, it aims to find universal principles. Even the original DW has much removed here. This may mean that GR as defined here is narrower than others assume; but by clarifying a line between the universal and the specific it may help illuminate in other DW studies just what is core (and generalisable) and what may not be.

The minimal 2-variable model

DW is an Alife-style model first presented by Lovelock in 1982 (published as Lovelock, 1983). The best-known early citation is here referred to as W&L (Watson and Lovelock, 1983). The minimal version here, based on Harvey (2004), radically simplifies the original DW of W&L. We start with a single 'bio-variable' *B*, and a single 'enviro-variable' *E*. They take values within finite ranges that we choose to scale as [0.0,1.0]. Together they form a dynamical system, where each affects the rate of change of the other via functions *H( )* and *Z( )*, parameterised by *P* as described further below:

$$\mu \, {}^{dB}\!/_{dt} = H_P(E) - B = H(E+P) - B \qquad (1)$$

$$\nu \, {}^{dE}\!/_{dt} = Z(B) - E \qquad (2)$$

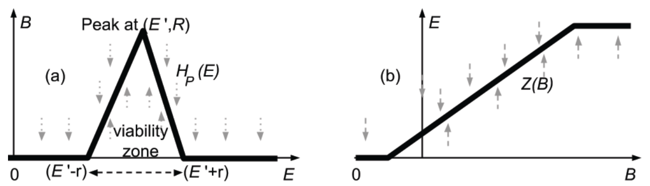


Figure 2. Example nullclines (a) $B=H_P(E)$ and (b) $E=Z(B)$. Directions of dB/dt and dE/dt are indicated by arrows.

μ and ν (that we shall typically set to 1) moderate the rates of change of B and E towards the 'nullcline' endstates of $B=H_P(E)$ and $E=Z(B)$. If and when both endstates are reached simultaneously, in the absence of noise further change would cease; we shall be considering the system in the presence of noise that will shift the system from unstable to stable equilibria. Such equilibria points can be seen graphically as where the nullclines intersect; the form of these nullclines is crucial to the model, determining which equilibria are stable and which unstable.

We choose $H_P(E)=H(E+P)$ as a parameterised 'hat-shaped' function that is zero outside some constrained 'E-viability-zone' within which it rises to a peak (Fig. 2a); parameter P shifts the hat left or right. Here we show a piecewise linear 'witches hat'; most of the results that concern us are insensitive to the precise form of the hat. With the peak value R of the hat at $E=E'$, and viability 'radius' r, we have:

$$H(E)=R.max(1-abs(E-E')/r,0) \qquad (3)$$

We choose for illustration (Fig. 2b) $Z(B)$ as, in the region where it affects matters, a linear relationship based on $(Q + sB)$; where Q is a fixed default value for E when B=0, and s is the gradient of slope (positive or negative). However this linear form is truncated below and above at E = 0 and 1. This results in a 'zigmoid' or piecewise-linear sigmoid:

$$Z(B)=min(max(Q+s.B,0),1) \qquad (4)$$

We may flip Fig. 2b, and overlay on 2a so that the B-E axes now coincide, as shown in Fig. 3, $Z(B)$ now shown dashed. These nullclines intersect at steady states, stable or unstable.

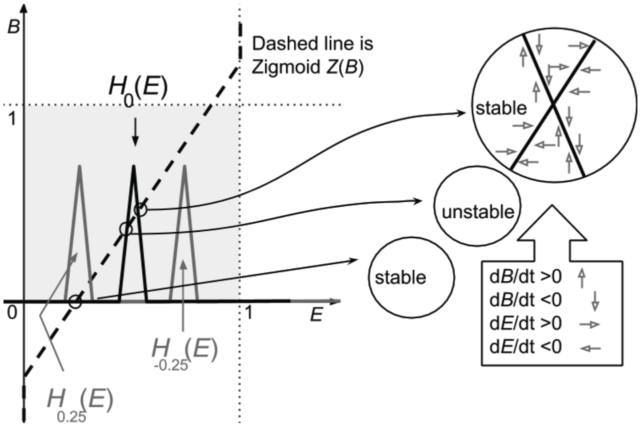


Figure 3. As the hat-function $H_P(E)$ shifts left and right with different P values, there are between 1 and 3 steady states. $H_0(E)$ has 3, circled; examination of the arrows representing dB/dt, dE/dt, indicate they are stable/unstable/stable.

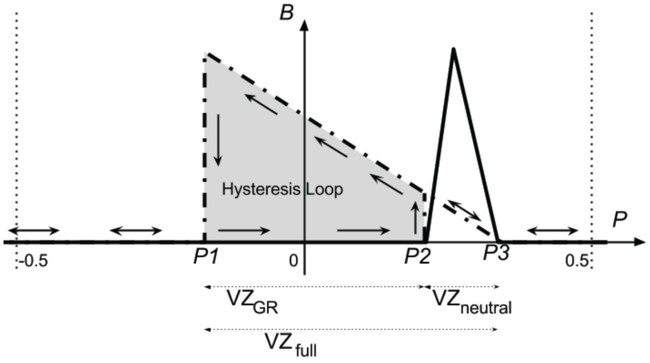


Figure 4. Values of B for the stable steady states of the BE system as P varies. Note that the axes are now B against P.

When parameter $P=0$, examination of the signs of dB/dt and dE/dt (Fig. 3, for $H_0(P)$), shows that the rightmost of 3 steady states is stable; likewise the leftmost is stable. The steady state between these on the nullcline is necessarily unstable. Hence if we assume there is noise in the system such that it leaves the unstable one, after transients the only (stable) steady states to be seen are as shown: one with B zero, one with B positive.

We plot these stable steady states in Fig. 4, as P is varied. For $P1<P<P2$ there are 2 possible values for B, providing a hysteresis loop: as P is increased from low values, the lower part of the loop will be followed, jumping up at $P2$; as P decreases from high values, the upper part of the loop is followed until a jump down at $P1$ (corresponding to where the nullclines are tangent to each other). We define Viability Zone VZ_{full} (in Fig. 4 from $P1$ to $P3$) as the region of P-space for which there is at least one stable steady state with B>0; i.e. here we consider the *upper* limb of the hysteresis loop.

The Neutral Comparison

The hat-function shown in Fig. 4 represents the values taken by B for the (single) steady state of a 'neutral' version of the system; here we follow W&L. This is what would happen if all the effects of B on E were nullified, i.e. if $Z(B)$ was replaced by $Z(0)$ as indicated by a hypothetical neutral 'switch' in Fig. 1, thus breaking the $B \rightleftarrows E$ circuit. This results in a viability zone $VZ_{neutral}$, where B>0, between $P2$ and $P3$. This width is 2r, the same as the basic width of $H(P+E)$, see eqn 3; but here it is P that is varying rather than E. We may note that $VZ_{neutral}$ corresponds exactly to VZ_{full} except that it is based on the *lower* (B=0) limb of the hysteresis loop rather than the *upper* limb.

In this example, we can see that VZ_{full} covers the zone $VZ_{neutral}$ and extends further. $VZ_{full} = VZ_{neutral} + VZ_{GR}$, where the latter 'Gaia-Regulated' VZ corresponds exactly to the hysteresis region $P1<P<P2$. We use the existence of such a non-empty VZ_{GR} as the basis for defining Gaian Regulation.

Gaian Regulation

We define GR as occurring whenever there is a zone of Perturbation-space within which a biotic variable B *can* be viable (B>0), despite a neutral version of B (where its effects are nullified) being non-viable (B=0); in other words, where there is this 'fragile' form of bistability with both upper and lower (zero) limbs of a hysteresis loop. We have demonstrated this in the simple example above, where the functions and parameters were chosen so as to highlight this. We chose a narrow hat-function (small r in eqn 3), and a high value for s, the slope of the zigmoid (eqn 4); such choices tends to make $\mathrm{VZ}_{\mathrm{GR}}$ more prominent for illustrative purposes. We note that this definition broadly corresponds to the implied definition in W&L, where B refers to daisies. E to temperature and P to Solar Luminosity. W&L recognised the hysteresis from the start; their Figure 1a clearly shows what is here called $\mathrm{VZ}_{\mathrm{neutral}}$ for a 'neutral daisy' (that has the same albedo as the ground, i.e. its effects are nullified). Elsewhere in W&L's Figure 1 they show what is called here $\mathrm{VZ}_{\mathrm{full}}$ for various combinations of daisies, expanding the viability range beyond $\mathrm{VZ}_{\mathrm{neutral}}$.

Sceptics will of course suspect that these simple examples are cherrypicked to show the phenomenon of GR, and that different examples may show some reverse anti-Gaian effect. In the second part of the paper we show otherwise; within a very broad range of dynamical systems of any size, we show that for any arbitrary variable B, $\mathrm{VZ}_{\mathrm{neutral}} \subseteq \mathrm{VZ}_{\mathrm{full}}$. In other words if the effects of B on its environment are such as to influence its own Viability Zone in any direction, this influence can only be in a positive direction, expanding the range of viability and thus displaying GR. Not lucky Gaia (Kirchner, 2002), but inevitable Gaia.

Various Fallacies and Confusions

Before we get there, we can use the present very simple model, loosely equivalent to black daisies in W&L's DW, to illustrate some commonly held fallacies about Gaia Theory.

The Misattributed-Feedback Fallacy. This is the all-too-easy error of calling, for instance, the effect of E on B a 'negative feedback' rather than a 'negative effect' (or positive, as the case may be). Feedback, -ve or +ve, is a property of a complete ***feedback circuit*** (-fc or +fc) of effects $A \rightarrow B \rightarrow C\ldots \rightarrow A$ and so cannot be attributed to any single effect $A \rightarrow B$. This mistake is made time and again, even in W&L where 'changing the direction of effects' is confused with 'changing the direction of feedbacks'. The present author has erred likewise (e.g. in Harvey, 2004). Although it is strictly true that changing the direction of a single effect in a feedback circuit will – *if* nothing else changes – reverse the sign of that circuit, in the current context there is *always* a consequent further change to counteract this. This sloppiness in language hence leads directly to …

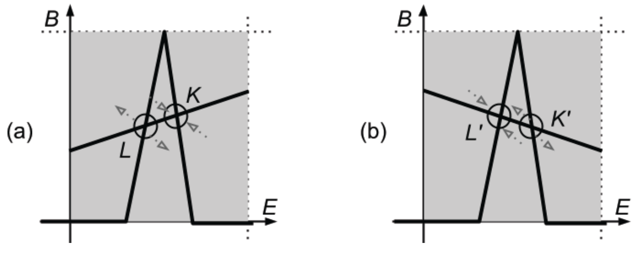


Figure 5. (a) Corresponds to Fig. 3, with -fc at K; this relates to 'black daisies' with a +ve effect. (b) If the slope is reversed ('white daisies'), K becomes K' (+fc) but also L becomes L' (-fc). The stable steady state does not 'disappear' but rather 'shifts elsewhere'.

The Missing-the-point Equilibrium Fallacy (aka Lucky Gaia). Confusing effects with feedbacks misleads the unwary into believing that since, e.g., a +ve effect is associated with +fc in one part of phase space, it will be associated with +fc in other parts of phase space. In Fig. 5 we have a counter-example: B has a +ve effect on E that contributes to a +fc at unstable equilibrium L, yet to a -fc at stable equilibrium K.

As indicated in Fig. 5, although swapping the sign of an effect does turn -fc into +fc, it simultaneously turns any neighouring +fc into a new -fc. So state of the dynamical system, in the presence of any noise, will automatically shift through phase space towards another point. The 'Lucky Gaia' criticism takes it as a matter of luck (dependent on directions of +ve or -ve effects) whether a specific steady state is stable or unstable; but one such point becoming 'unlucky' means the next one will become 'lucky'.

The Optimising Gaia Fallacy. It should be clear from the simple example that nothing there can be identified as being 'optimised'. Under Gaian regulation, the peak value of B (seen at $P=P1$ in Fig. 4) is the same as it was without GR. Further, in the simple example shown, this peak value is right at the tipping-point where regulation is about to be lost. Within $\mathrm{VZ}_{\mathrm{neutral}}$ the value of B is mostly decreased. The Viability Zone is typically *increased* under GR, but that does not translate to its being *optimised* – optimised with respect to what? Suggestions to the contrary, such as:

> We have since defined Gaia as a complex entity involving the Earth's biosphere, atmosphere, oceans, and soil; the totality constituting a feedback or cybernetic system which seeks an **optimal** physical and chemical environment for life on this planet. (Lovelock, 1979)

are better replaced by formulations such as:

> The Gaia hypothesis ... the atmosphere, the oceans, the climate and the crust of the Earth are regulated at a state **comfortable** for life… (Lovelock, 1979) [stresses added]

and **habitable** is more appropriate still (Kirchner, 2003).

The Setpoint Fallacy. One probable motive for the Optimising Fallacy is that conventional Regulators use a predetermined setpoint, with -ve feedback bringing a perturbed system back to that point. This immediately provides something to be optimised – 'distance from setpoint'. But GR is interestingly different from this, and has no fixed setpoint (c.f. 'rein control' below); as Perturbing Parameter P changes, so do the positions of equilibria in phase space.

The Beneficial/Harmful Confusion. At a stable steady state, as at K in Fig. 5, the -fc is *beneficial* in this sense: a change in the external Perturbation P will be compensated for by the feedback circuit tending to reduce the effect of that change. This is no more or less than the Le Chatelier Principle (Le Châtelier and Boudouard, 1898), well known and accepted in chemistry and economics. Nevertheless, around steady state K the effect that B has on E (+ve in this example), is actually *harmful* to B, in that an increase in E leads to a consequent decrease in B. The +ve effect of B on E (in this example 'black daisies'; the argument takes the same course with a -ve effect with 'white daisies') can be considered to be simultaneously (a) apparently 'G-beneficial' in promoting GR by forming part of the -fc that allows this stable state with nonzero B to exist and (b) L-harmful, i.e. locally harmful to small variations in the B-effect on E. This is not the contradiction it first seems to be, since the G-beneficial describes the feedback rather than the effect (see Misattributed Feedback Fallacy). Nevertheless this has led to much confusion. E.g. the subtleties of beneficial/harmful in a Gaian context are not noticed in these quotations from a critic of Gaia theory:

> Coupling between the biosphere and the physical environment can potentially give rise to either negative (stabilizing) feedback, or positive (destabilizing) feedback, and the consequences of this feedback can potentially be either beneficial or detrimental for any given group of organisms. (Kirchner, 2002)
>
> Which brings us to Daisyworld. The Daisyworld model assumes that traits that benefit the environment also give an individual a reproductive advantage over its neighbors. (Kirchner, 2002)

The Evolution and Gaia Fallacies. This same confusion is also displayed by advocates of Gaia theory, as for example:

> In Daisyworld, natural selection is directly linked to environmental effects such that what is selected for at the individual level is beneficial to the global environment. (Lenton and Lovelock, 2001)

Natural selection refers to Darwinian evolution, and this statement is, as a general principle, entirely wrong. Darwinists know how easy it is to show counter-examples; this feeds their distrust of Gaia Theory. Later in the same paper a possible cause for this confusion is suggested where "what is selected for at the individual level is also beneficial at the global level" is offered as a rephrasing of "organisms alter their immediate (local) environment and the global environment in the same way". There is a sense of 'selection' in which black daisies within DW may be said to be 'selected' under GR when the sun is 'too cold', and white daisies 'selected' when it is 'too hot'; different organisms flourish in different environments. But emphatically this is not *natural selection* in any Darwinian sense, of variant organisms being preferentially selected in the same environment. As a general principle we may observe the opposite: what is G-beneficial to the global environment will be naturally selected *against* if Darwinian evolution occurs within the timescales of our model.

Above we noted that the effect of B on E was locally L-harmful. Any positive variation in this effect – for instance a mutation that caused a black daisy to have a slightly stronger tendency to absorb heat from the sun, thereby locally increasing temperature E – would actually (around a stable equilibrium such as K) *decrease* the amount of B. Such variations are broadly equivalent to changing the slope s in eqn 4. Hence, if the consequences were felt by the mutant B more than by its neighbours, that mutation would be selected against. Such selection pressure on albedo-changing mutations in black daisies tends (in the absence of constraints) to drive them towards neutral; likewise in white daisies. So evolution on such biotic effects ($B \rightarrow E$) can, under plausible circumstances, be expected to decrease and ultimately eliminate (the need for) GR.

Many critics of Gaia Theory assume that GR must have arisen through evolution, since it appears to be in some sense adaptive and hence seems to need some explanation for its origins. The previous Kirchner (2002) quote, discussing 'reproductive advantage', buys into this idea, and of course Dawkins' many criticisms of Gaia (Dawkins, 1982) address the same issue. In fact evolution is neither required for the display of GR – the present simple model makes no reference to evolution – nor, since as we argue that it does not depend on luck, does it require an evolutionary explanation for its origins. We return to further discussion of evolution below.

The Stability-Unlikely Fallacy. It is fallaciously believed by many that as a dynamical system becomes more complex, it is increasingly unlikely to have any stable steady states at all (May, 1972). The errors in May's analysis, as applied to nonlinear systems, have been pointed out elsewhere (Harvey, 2011); one error relates directly to the Missing-the-Point Fallacy. We demonstrate the contrary in the next section, in particular the inevitably of stable equilibria in our systems.

Two Reins and More

The minimal example used so far to display core GR has had just one type of biota B. This may puzzle people who think that DW relies on there being (at least) two daisies, black and white; though from the start W&L observed such regulation with a single type of daisy (e.g. see their Figure 1b).

Black daisies can G-regulate temperature when otherwise it would be too cold; it needs white daisies, or their equivalent, to G-regulate when it is too hot. Though a single environmental variable is being regulated, it needs two separate pathways for regulation in both directions. This matches exactly with Clynes' (1969) observations of 'rein control' (or 'unidirectional rate sensitivity') in biological homeostasis. Each rein of a horse can only pull in one direction, not push; hence for control in both directions two separate reins are needed. When doing so, the circumstances under which they may tend to cancel each other out, rather than complement each other, are discussed in Harvey (2004).

Saunders et al. (1998) were the first to relate rein control to DW. However they were largely focussed on circumstances where zero steady-state error may be achieved – that necessarily implies error with respect to some setpoint. To that end they added a version of integral control (that employs a signal related to time integral of error) to produce what they call 'Integral Rein Control'. To clarify, the original Clynes (1969) version of rein control, as used here and in Harvey (2004), is just plain rein control with no 'integral' element. As such, it has no need for the concept of a setpoint or of error.

The concept of rein control explains why two bio-variables are needed to control an enviro-variable in two directions. Further, it explains why, with many bio-variables but a single enviro-variable (McDonald-Gibson et al., 2008), at the core there is a dynamic equilibrium between those bio-variables that are 'pulling one way' and those 'pulling the other way'.

Extending to several enviro-variables is interesting and challenging (Dyke and Weaver, 2013); a start has been made there on analysing phase-portraits of two- and three-enviro-variable systems. But here we now prove a theorem valid for arbitrary numbers of variables in a very general model.

Gaian Regulation Theorem

The minimal model presented above is a simple example of the general case, and so may be useful in guiding interpretation. We consider *m* bio-variables (*bvars*), **B**=(b_1,..,b_m); *n* enviro-variables (*evars*), **E**=(e_1,..,e_n); and a set of *n* parameters, or external perturbations, associated with the *n evars*, **P**=(p_1,..,p_n). **B** and **E** form a coupled dynamical system, parameterised by **P**. As before, we shall be examining the stable steady states of this system for different **P**, and in particular which regions of **P**-space allow one or more *bvars* from **B** to be viable. All variables are finite and bounded. This means we can rescale each variable in **B** and **E** to lie within the range [0.0,1.0] at all times.

For many purposes *bvars* and *evars* will be treated identically from a mathematical perspective. There are just two differences. Firstly only **E** is directly influenced by exogenous influence from **P**; indirect influence on the *bvars* is only via the *evars*. Secondly, for the term 'viable' to make any sense when describing *bvars*, there must be a contrast with 'non-viable' (i.e. stable steady state zero); for each *bvar* there must be both viable (>0) and non-viable (=0) regions.

It will be useful to introduce the notation $\mathbf{B_{-i}}$ to refer to all the *bvars* excluding the *i*th; similarly $\mathbf{E_{-j}}$ excludes the *j*th *evar*. Our dynamical system is then defined by *m*+*n* equations of this form, using any continuous functions h_i() and f_j() at all that obey the bounding constraints below:

$$\mu_i \, {}^{db_i}\!/_{dt} = h_i(\boldsymbol{B}_{-i}, \boldsymbol{E}) - b_i \qquad \text{for } i\text{=1 to } m \ (5)$$

$$\nu_j \, {}^{de_j}\!/_{dt} = f_j(\boldsymbol{B}, \boldsymbol{E}_{-j}, \boldsymbol{g}_j(\boldsymbol{P})) - e_j \qquad \text{for } j\text{=1 to } n \ (6)$$

The μ and ν moderate the rates of change, and we shall typically set these to 1. The g_j(**P**) specify differently weighted subsets of **P**; any weighting is permissible. In turn these generate *m*+*n* nullclines of the form:

$$b_i = h_i(\boldsymbol{B}_{-i}, \boldsymbol{E}) \qquad \text{for } i\text{=1 to } m \ (7)$$

$$e_j = f_j(\boldsymbol{B}, \boldsymbol{E}_{-j}, \boldsymbol{g}_j(\boldsymbol{P})) \qquad \text{for } j\text{=1 to } n \ (8)$$

Bounding Constraints. The important constraints that we put on each h_i() and f_j() are that they must be continuous and lying within the range [0.0,1.0]. On translation to the real world, this reflects the fact that we only consider variables that are bounded below and above (e.g. for a species, it is bounded below at zero and above somewhere before the biomass exceeds the mass of the planet); and we may rescale all variables to lie within [0.0,1.0]. The consequence is that, although such functions may be defined for arguments outside that interval, they can only return values within it. Hence all the nullclines so defined intersect each other within the unit hypercube (defined so as to include its boundary). It follows that, from any starting position in phase space within the unit hypercube, there can be no trajectory leading out of it. When counting steady states, we need only search within this; no variables will 'shoot off to infinity'.

Counting Steady States

For this preliminary stage, we just consider the *m*+*n* nullclines, with no need to distinguish between *bvars* and *evars*; we emphasise the generality of this result this by relabelling both here as **X**=(x_1, x_2, …x_{m+n}). We shall prove by induction Hypothesis 1 that: regardless of the number of variables, there are 2*q*-1 steady states, for some *q* ≥ 1, of which *q* are stable and *q*-1 unstable. We start by proving the result holds for 2 variables, and then show that the result still holds as we add one extra variable; by iterating we can reach any number of variables.

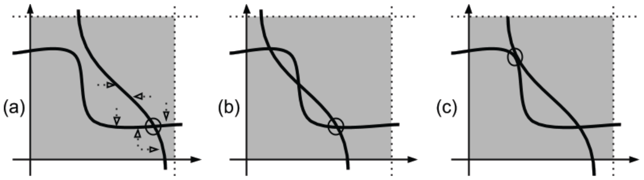


Figure 6. (a) The N-S nullcline must cross the W-E nullcline at least once; (b) if more, an odd number, provided (c) tangents are treated as a coincident pair of intersections.

Fig. 6 shows the 2 variable case (x_1, x_2); we consider each nullcline as starting and ending outside the unit box. They cannot intersect outside the box (because they are enclosed by the box boundaries), and the only intersections we need to count are within the box (including its boundaries). The nullcline heading north from the south edge of the box starts *below* the west-east nullcline and finishes, outside the north edge, *above* the west-east nullcline. Since each crossing toggles between *above* and *below,* there must be 2*q*-1 steady states, where the nullclines intersect, for some *q* ≥ 1. In Fig. 6a, the arrows around the circled intersection indicate the directions of dx_1/dt and dx_2/dt associated with the nullclines, demonstrating this steady state is stable.

In Fig. 6b, the circled intersection is clearly locally similar to that in 6a, hence will also be stable. A similar argument shows that the intersection in 6b furthest from that circled must also be stable; the central intersection can only be an unstable steady state. Since intersections must always alternate between stable and unstable along any nullcline, in the general 2-variable case we must indeed have 2*q*-1 steady states, for some *q* ≥ 1, of which *q* are stable and *q*-1 unstable. This is our hypothesis satisfied for just 2 variables. We note that since the nullclines are continuous functions of their variables and parameters, any smooth shift in these will result in a smooth movement through phase space of these steady states; except that where (Fig. 6c) tangents are created or lost, pairs of steady states (stable + unstable) appear or disappear; tangent intersections are counted double (Fig. 6c). In bifurcation theory this counts as a pitchfork bifurcation.

How do we extend Hypothesis 1 to 3 variables, where we are looking at 2-D nullclines intersecting in a 3-D unit cube? We note that if we consider a planar slice across the 'bottom' of the cube, where the new third variable x_3=0, the desired property holds true in x_1-x_2 space: there will be 2*q*-1 points on that plane representing steady states. As we move the planar slice 'upwards', i.e. smoothly increasing x_3, these points indicating steady states will likewise shift smoothly, with possibly pairs of new points being created (and then separating) or coming together (and then vanishing). The result will be the intersection of the x_1-nulline and x_2-nullcline in the form of a set of lines (minimum 1) that span the gap between the x_3=0 'bottom' plane and the x_3=1 'top' plane. The steady states of the 3-variable system will be those points where this set of lines cross the x_3-nullcline, that lies somewhere between the top and bottom of the cube.

If such lines cross the x_3-nullcline just once, we necessarily have the required $2q$-1 intersections; if, via any folds in such lines and/or in the x_3-nullcline, any such line crosses more than once, it must add an even number of crossings. Hence the required $2q$-1 intersections still applies. We note that, through reasoning similar to that applied in the 2-D case, any such intersection that is adjacent to the corner of the unit cube is necessarily stable. Likewise, successive steady states alternate between stable and unstable. Hence we can extrapolate from the 2-D case to the 3-D case, where Hypothesis 1 still holds: there are $2q_3$-1 steady states, for some $q_3 \geq 1$, of which q_3 are stable and q_3-1 unstable.

Though it is more difficult to visualise in higher dimensions, we can make exactly the same arguments in progressing from 3 variables to 4 variables, and thus iterate further to any arbitrary number of variables. Since in considering n-dim space we already have the result for (n-1)-dim space, we are each time considering a number of threads ($2q_{(n-1)}$-1) starting outside the 'bottom' (n-1)-hyperplane of the n-hypercube, and progressing continuously until exiting at the 'top'. These threads correspond to intersections of (n-1) nullclines (that are hyperplanes) and must at some stage pass through, on their way from bottom to top, the intervening nth nullcline-hyperplane. The only way any continuous thread may terminate as one travels up, other than exiting at the top, is for the equivalent of tangent collisions to be made or broken. As we have already seen, these add (or subtract) pairs of intersections that separate (or come together). Thus any one thread is ultimately continuous from bottom to top and crosses the intervening nullcline an odd number of times, minimum once. The pattern of the Hypothesis is carried over to the next dimension, now with $2q_n$-1 steady states, q_n of these stable.

This preliminary stage means we can guarantee the existence of stable steady states, typically along with unstable ones, regardless of the complexity of our system; the bounding box is in its entirety a basin of attraction for stable steady states. We need this result to continue our proof.

Comparing Viability Zones. We extend the VZ definitions from this minimal context to the more general domain.

We select any arbitrary *bvar* b_i to be considered as the focus of interest, and define the VZs associated with it. Each of these zones will be a zone in **P** space or a subset of **P** space.

For each possible value of **P** we can find in principle all the steady states of the full set of equations; we have proved above that such steady states exist within the unit hypercube. If (for a specific value of **P**) there is any stable steady state with x_i>0 then by definition this specific value of **P** is within VZ_{full}. We next generate VZ_{neutral} by doing a similar exercise but with x_i 'neutralised'; within all the other n-1 equations, x_i is replaced by zero, thus having no effects, +ve or -ve, on any of the other variables. As with the minimal 2-variable case, we have VZ_{neutral} and VZ_{full} each defined as zones within Parameter space **P**. Can we prove anything about their relationship? Surprisingly, yes – and easily.

Reductio ad Absurdum Proof of Theorem

The Hypothesis 2 that we now wish to prove is $VZ_{neutral} \subseteq VZ_{\text{full}}$. Suppose this was false. Then there is at least one point in **P** space that lies within $VZ_{neutral}$ but outside VZ_{full}. We consider such a parameter set.

Since it is outside VZ_{full}, it follows that there all stable steady states of the system require x_i to be nonviable, i.e. x_i=0. But this is exactly the condition we impose when we neutralise x_i so as to define $VZ_{neutral}$; hence we have shown that this point in **P** space must lie outside $VZ_{neutral}$. Assuming the Hypothesis to be false has produced a contradiction. QED.

A less rigorous, but perhaps more comprehensible, explanation would be: if this point in parameter space produces a non-viable x_i in the full (un-neutralised) system, then it will definitely behave the same way in the neutralised version (where x_i is constrained to be zero). We have proved $VZ_{neutral} \subseteq VZ_{\text{full}}$.

This means that the relationship between these VZs may be equality, if x_i has no effects on its own VZ; this will certainly be the case if VZ_{neutral} covers all of parameter space. But *if* it has any effect at all on its VZ, this can only be to increase its size. This means we can define a 'Gaia Regulated' VZ_{GR}, such that $VZ_{\text{full}} = VZ_{\text{neutral}} + VZ_{\text{GR}}$. We note again, as above, that ideal conditions for VZ_{GR} to be significant in size include VZ_{neutral} being small and constrained; e.g. if x_i refers to some complex organism with tight constraints on its viability.

Corollaries. The original minimal DW example had the various VZs as continuous intervals in the space of a single parameter. This generalised version has no such constraints; not only can any number of variables be included in the viability conditions, but also disjoint VZs are equally valid.

Because the result $VZ_{\text{neutral}} \subseteq VZ_{\text{full}}$., for any variable x_i is derived from such a general system, we need not assume that x_i necessarily refers to a single biotic (or indeed environmental) variable. It could for instance apply to a variable defined as 'black AND white daisies', likewise 'black OR white daisies', using logical AND and OR. Indeed the Gaian Regulation Theorem will apply to any entity, including a complete ecosystem, that (a) conforms to any version of equations (5) and (6), (b) can be assessed for viability as some exogenous parameter set influencing the environment varies, (c) has some effect on that same environment, thus (d) forming feedback circuits with stable steady states.

Why does the Theorem Work?

The theorem is very general in application. It is instructive to see just which minimal but necessary constraints provide the interesting results.

Firstly, the form of all the equations used, both for bio-variables (eqn 5) and enviro-variables (eqn 6), is such as to be naturally interpretable in terms of continuous nullclines (eqns 7 and 8). This follows the precedent set by most previous analyses. We consider the steady states of such systems.

Secondly, there is an implicit division of timescales into 3 ranges. Explicitly, the only ones expressed in the equations are μ and ν which provide the timescale within which the system finds its way to stable steady states. Implicitly, there is the faster timescale of small 'thermal' noise that we assume will dislodge the system from any unstable steady state; and the slower timescale of any changes in the exogenous parameters **P**. We assume such parameters stay fixed long enough for stable steady states to be reached. We return to this below.

Thirdly, finite bounds are put on the nullclines, and we choose to rescale the variables such that nullclines lie within [0.0,1.0]. This immediately means that they can only intersect within the unit hypercube, and guarantees us stable equilibria (that the system has time to reach), as well as any unstable ones. As shown above, we can relate numbers of stable states to unstable ones, and observe that they alternate along nullclines. This is the trick that was missed in previous studies of similar complex systems that considered only linear ones (Gardner and Ashby, 1970), concluding that instability became inevitable as numbers of variables increased; and in studies that claimed inaccurately to extend such linear results to the nonlinear case (May, 1972). Errors in the latter studies have been previously pointed out (Harvey, 2011); it is a classic example of the Missing-the-Point Fallacy.

Fourthly, the concept of viability, states of affairs where a bio-variable can be nonzero, is central. This comes directly from the cybernetic era that provided a context for the origins of DW. Ashby (1952) introduced the similar idea of 'essential variables' to his analysis of homeostasis and homeostats; as an aside, his Chapter 20 on 'Stability' provides the explicit motivation for Gardner and Ashby (1970) – but misses out what nonlinearity brings to the stability table. The key aspect of viability here is its potential fragility, where it may be found and lost. As we saw earlier (Fig. 4), VZ_{GR} relates to zones of hysteresis where B may have either +ve or zero values. The hysteresis can be interpreted as a symptom of fragility, it may be a 'struggle' to recover from a loss of viability. The boundaries to viability are crucial to GR.

Fifthly, GR is defined in terms of zones of Parameter-space and not zones of enviro-variable space. The original DW model was clear about this; for instance W&L, in their Figure 1, illustrate viability as their parameter of Solar Insolation varies.

Sixthly, also explicit in W&L (e.g. their Figure 1a), is the comparison made with a 'neutral' version (in their case a neutral colour of daisy) that generates $VZ_{neutral}$. Basically, VZ_{full} identifies the P-zone where B is viable ***including*** any hysteresis loop (i.e. counting the upper, viable part of such a loop), whereas $VZ_{neutral}$ identifies the same but excluding any hysteresis loop (by in effect counting the bottom, zero-valued part). VZ_{GR} is defined as the difference, the 'fragile zone' of the hysteresis loop. This allows the remarkably simple yet powerful Reductio ad Absurdum proof used above. Indeed with this insight it becomes clearer why there is nothing that could count as the inverse of VZ_{GR}, there is no possibility of 'anti-Gaian-regulation' as assumed by lucky-Gaia proponents.

Tipping Points. The boundaries of VZ_{GR} are associated with discontinuous jumps in the hysteresis loop, as seen at P1 and P2 in Fig. 4.; one will be experienced as P decreases, the other as P increases. Whereas the P2 boundary is shared between VZ_{GR} and $VZ_{neutral}$, the other P1 boundary derives from where the enviro-variable nullcline (here a zigmoid) is tangent to the bio-nullcline (here a hat function). Given the use of a witches hat, it so happens that this coincides with the peak value of bio-variable *B*. More generally, any form of hat may be used, and as seen in Fig. 7, the tangent associated with the tipping point need not be anywhere near the peak.

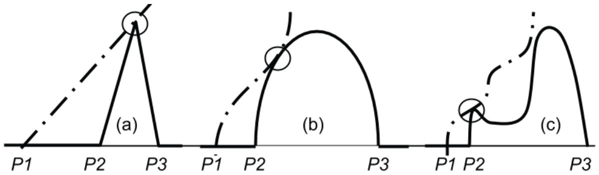


Figure 7. (a) P1 is associated with the tangent (circled) at the peak of a witches hat. More generally, (b) and (c), such tangents need not be at the peak of the hat.

Gaia and Evolution

The basic DW model contains no element of Darwinian evolution, yet displays GR; so GR does not require evolution for *its ongoing operation*. Further, given the fallacious reasoning behind 'lucky Gaia', GR is not some improbable phenomenon that needs adaptive explanations like evolution to explain *its origins*. It arises naturally and inevitably whenever there is some 'fragile' system with the appropriate kind of hysteresis loop. In so far as natural evolution generates living systems that are indeed fragile in that sense, viable only within some ecological niche, it generates the raw material for GR – which then arises by definition, rather than through some further mystery that needs explaining.

Evolving the effects of *B* on *E*. We have seen above (in discussion of the Beneficial/Harmful Confusion, the Evolution and Gaia Fallacies) that there is an opposition between evolution and GR in this sense: when a biotic trait has an environmental effect that contributes to a GR feedback circuit, it is L-harmful and selection would tend (in the absence of constraints) to reduce or eliminate it. A simple interpretation of this is: it is in an organism's interests to evolve so as to *decrease* its fragility within its ecological niche, and in so doing so it inevitably *reduces* the scope (and need) for GR.

Evolving the effects of *E* on *B*. We may also consider the possibility of evolution evolving biota so as to alter their susceptibility to environmental conditions. It can readily be shown (Robertson and Robinson, 1998) that if daisies in DW have unconstrained capacity to evolve so that their optimal growth temperatures (the 'peak' of the hat-function) match the prevailing temperature, then likewise this will *reduce* the scope for GR and ultimately eliminate it. A response (Lenton and Lovelock, 2000) broadly agrees, whilst asserting that in real systems where physical constraints set bounds to the limits of such adaptation, GR can take over when Darwinian evolution runs up against such constraints. Again we see GR and Darwinian evolution as complementary not antagonistic.

Incorporating Evolution inside this model. It is of course possible to incorporate the dynamics of evolution directly within this model, provided the 'terms and conditions' (t&cs) are respected. This typically requires the evolutionary dynamics to run to steady state before any parameter *P* is changed. Traits that are evolvable will typically (a) not be directly affected by parameters and (b) be viable or non-viable at steady state; hence they fit the requirements to be labelled as *bvars*. If eqns 5 and 6 can be tailored to reflect the evolutionary dynamics, then the GR Theorem will apply to such traits just as with any other *bvar*.

A Vicious Circle in Gaia/Evolution debates. DW was invented without reference to evolution (there was no need). Darwinians misunderstood, assumed GR could only be evolved yet it was susceptible to 'cheats'; hence they declared it impossible. DW modellers knew this was not so, and created DW extensions to demonstrate this. Unfortunately they took 'anecdotal' specific examples and elevated them into general principles without justification. Darwinians easily found 'anecdotal' counter-examples to these general principles, and asserted the opposite. The cycle of misunderstanding and suspicion continued.

For example, some studies combining evolution and DW (e.g. Lenton, 1998) mix the timescales by having parameters changing simultaneously with the evolutionary dynamics. The results are valid for those specific choices, but are in this context 'anecdotal' since we have no principles to decide how far we can generalise them. The GR Theorem presented here will not extend to such results, but is fully generalisable within its own carefully stated constraints, its t&cs.

Discussion

This paper is deliberately fact-free, it is mathematics rather than science. The analysis has focused on abstract models that come with t&cs. Any real world lessons depend whether real phenomena do indeed match the t&cs, and this is outside the scope of this paper. Nevertheless, it is hoped that some of the ideas and intuitions here may be useful tools for scientists. The broad generality of the results make it tempting to try and fit this model-template to the world; however we may warn that one of the most challenging t&cs to observe may be the strict separation of timescales. The results here depend on the parameters P being maintained fixed long enough for the $B \rightleftarrows E$ dynamics to reach steady state.

Going out on a limb. An appropriate metaphor for the fragile nature of GR is that of going out on a limb – on the top limb of a hysteresis loop, from the safe tree trunk of $VZ_{neutral}$. The mathematical results here should be safe, but we may speculate what future directions may be promising. One such is going beyond steady state phenomena to metastable states.

Slower timescale changes may alter, even eliminate the safe zone of $VZ_{neutral}$; to rephrase Wittgenstein and his ladder, throwing the tree trunk away after one has climbed up it. Where you are on a hysteresis loop depends on historical contingencies of how you got there, hence changes at multiple timescales may eliminate possibilities of going back to safety. Darwinian evolution of course introduces new timescales.

This may present a picture of Life – and on different scales e.g. tornadoes, and planetary viability for biota – as potentially fragile existence on a limb of multidimensional hysteresis loops with no going back. There are strong echoes here of an autopoietic definition of Life (Varela et al., 1974); this is unsurprising, since autopoiesis and DW have shared origins in many ideas from cybernetics. Such fragility is of course balanced against the powerful forces of Gaian regulation providing the supporting limbs; "Gaia is a tough bitch", as Lynn Margulis commented.

Conclusions. This study shows Gaian regulation merely in a model, not the real world. Nevertheless we can debunk many widely held fallacies; critics of Gaia Theory are not justified in their appeals to these specific misunderstandings. GR is not 'lucky', it is inevitable (subject to the t&cs). The relationship between GR and Darwinian evolution is more subtle and complex than it is often misrepresented to be; their different roles may be seen as more complementary than antagonistic. Bounded physical variables imply the existence of stable steady states, and hence inevitable GR. However this GR is not 'optimising', not even really a 'comfortable' Gaia; perhaps best called 'habitable' Gaia, it fits nearest to what Kirchner (2003) calls 'biological feedback at the limits of habitability'.

The main mathematical lesson is that the curious circular logic of Gaia is full of surprises and challenges our intuitions.

References

Ashby, W. R. (1952). Design for a brain. Chapman and Hall.

Clynes, M. (1969). Cybernetic implications of rein control in perceptual and conceptual organization. *Ann. NY Acad. Sci.* 156:629-670

Dawkins, R. (1982). The Extended Phenotype. W. H. Freeman, Oxford.

Dyke, J. G. and Weaver, I. S. (2013). The emergence of environmental homeostasis in complex ecosystems. *PLOS Computational Biology* 9(5):1-9.

Gardner, M. R. and Ashby, W. R. (1970). Connectance of large dynamical (cybernetic) systems: critical values for stability. *Nature,* 228:784.

Harvey, I. (2004). Homeostasis and rein control: from Daisyworld to active perception. In Pollack, J. et al., editors, *Proc. 9th Intl. Conf. on Simulation and Synthesis of Living Systems, ALIFE9,* pages 309-314. MIT Press.

Harvey, I. (2011). Opening stable doors: complexity and stability in nonlinear systems. In Lenaerts, T. et al., editors, *Advances in Artificial Life, ECAL 2011,* pages 805-812. MIT Press.

Kirchner, J. W. (2002). The Gaia hypothesis: fact, theory and wishful thinking. *Climatic Change* 52:391-408.

Kirchner, J. W. (2003). The Gaia hypothesis: conjectures and refutations. *Climatic Change* 58:21-45.

Le Châtelier, H. and Boudouard O. (1898). Limits of Flammability of Gaseous Mixtures. *Bulletin de la Société Chimique de France (Paris)*, 19:483-488.

Lenton, T. M. (1998). Gaia and natural selection. *Nature* 394:439-447.

Lenton, T. M. and Lovelock, J. E. (2000). Daisyworld is Darwinian: constraints on adaptation are important for planetary self-regulation. *J. Theor. Biol. 206*:109-114.

Lenton, T. M. and Lovelock, J. E. (2001). Daisyworld revisited: quantifying biological effects on planetary self-regulation. *Tellus* 53B:288-305.

Lovelock, J. E. (1979). Gaia: a new look at life on Earth. Oxford University Press.

Lovelock, J. E. (1983). Gaia as seen through the atmosphere. In Westbroek, P. and de Jong, E. W., eds, *Biomineralization and biological metal accumulation,* pages 15-25. D. Reidel Publishing Company, Dordrecht.

May, R. M. (1972). Will a large complex system be stable? *Nature,* 238:413-414.

McDonald-Gibson, J., Dyke, J. G., Di Paolo, E. and Harvey, I. (2008). Environmental regulation can arise under minimal assumptions. *J. Theor. Biol.* 251(4):653-666.

Robertson, D. and Robinson, J. (1998). Darwinian Daisyworld. *J. Theor. Biol.* 195:129-134.

Saunders, P. T., Koeslag, J. H., and Wessels, J. A. (1998). Integral rein control in physiology. *J. Theor. Biol.* 194:163-173.

Varela, F. J., Maturana, H, R., and Uribe, R. (1974). Autopoiesis: the organization of living systems, its characterization and a model. *Biosystems* 5:187–196.

Watson, A. J. and Lovelock, J. E. (1983). Biological homeostasis of the global environment: the parable of Daisyworld. *Tellus* 35B:284-289.

Wood, A. J., Ackland, G., Dyke, J., Williams, H. and Lenton, T. (2008). Daisyworld: a review. *Reviews of Geophysics* 46(1):RG1001.

Conservation of Matter Increases Evolutionary Activity

Simon Hickinbotham Susan Stepney

Department of Computer Science, University of York, York YO10 5GH, UK,
sjh518@york.ac.uk

Abstract

We explore the hypothesis that adding conservation of matter to an artificial life system can increase its evolutionary activity, through experiments with the Stringmol artificial chemistry. Our first experiment examines the effect of varying the number of opcodes and finds a concentration which maximises the evolutionary activity of the system. The second experiment searches for the optimum *relative* concentrations of opcodes that maximises evolutionary activity: it finds increased evolutionary activity, a high diversity of opcode concentrations in each search run, and a different configuration of concentrations in separate search runs. The third experiment investigates the need for low concentrations of opcodes in high evolutionary activity, and finds that evo activity decreass when more of these particular opcodes are provided. We conclude that conservation of matter provides an important evolutionary pressure that can lead to more diversity and more evolutionary activity, and is therefore a desirable property for experiments in evolving ALife systems.

keywords: artificial chemistry; automata chemistry; evolutionary activity

Introduction: the role of matter in evolving systems

Biological evolutionary systems exist in the physical world and so they *must* conserve matter, yet virtual evolutionary systems are not obliged to conform to this constraint. What pressures does the conservation of matter place on an evolving system? Can an evolving system exploit this constraint? What advantages might it confer on synthetic evolutionary systems? Here we investigate this idea by examining the effect of imposing an additional constraint (limitation of material resources) on the 'evolutionary activity' generated by an automata chemistry.

We have previously demonstrated how a suitable artificial energy flux can lead to an evolutionary diversity "sweet spot": a low flux is too constraining to allow exploration the evolutionary landscape; a high flux provides no incentive to do so (Hoverd and Stepney, 2011). Can conservation of artificial matter provide similar advantages? Schneider and Sagan (2005) make clear the distinction between open energy flux and closed material recycling in ecosystems. Lones et al. (2013) show an artificial system where an analogue of conservation of matter allows evolved solutions to be more readily found. Fernando and Rowe (2007) have experimented with conservation of matter in a simple artificial chemistry, and found that the restrictions it imposes on a reacting system can drive it to self-organise. However, these ideas have not previously been applied to the domain of automata chemistries (Dittrich et al., 2001), in which the reactions between artificial molecules follows a program encoded in the structure of the molecule.

Stringmol (Hickinbotham et al., 2012) is an automata chemistry designed to explore intrinsic evolution *in silico*. It has demonstrated interesting behaviours such as hypercyles and emergent macro-mutations (Hickinbotham et al., 2010). In its basic form its execution rate is throttled by an externally imposed artificial energy flux, which provides selection pressure. Apart from in the work reported here, Stringmol is not constrained in how much "matter" it uses: its atomic opcodes are assumed to be available in arbitrarily large quantities.

Our central claim is that conservation of matter will increase evolutionary activity in an artificial system. We demonstrate this through three experiments. Our first experiment looks at the effect of having all opcodes being present in the same concentration, and searches for the absolute concentration that maximises evolutionary activity. Our second experiment searches for the optimum relative concentrations of opcodes that maximises evolutionary activity. Our third experiment takes evolved concentrations and determines whether high evolutionary activity is dependent upon low concentrations of key opcodes.

The structure of the rest of the paper is as follows. After providing some terminology, we provide some contextual material on conservation of matter in artificial chemistries (AChems). Next we discuss the modifications made to Stringmol to conserve matter. Then we summarise our measure of evolutionary activity used to quantify the differences made. We detail the three experiments, followed by a final section which discusses the results of our experiments in the context of artificial life.

Terminology

It is necessary to give clear working definitions of some of the concepts we discuss at this point as many different definitions exist in the literature. Throughout the remainder of this paper, the following definitions are used:

- **Matter:** in biology, the basic unit of matter is the atom. By analogy, matter in a virtual system is anything that can be regarded as the basic unit of the composition of the system.
- **Conservation of Matter:** the basic principle that matter cannot be made or destroyed in a closed system.
- **Opcode:** the 'atomic' element of an automata chemistry. It is the building block of the sequence of the molecule. Each opcode has a function associated with it, allowing the sequence to be interpreted as a program.
- **Molecule:** any collection of one or more opcodes that has been assigned as a molecule by a Stringmol reaction. The difference between an opcode in 'atomic form' and a molecule with sequence length of 1 opcode is that the latter has the reactive structures (pointers and flags) that allow it to be executed as a program.
- **Concentration:** The total number of opcodes present in a simulation. (This definition is applicable to systems that have no concept of space.)
- **Species:** a type of molecule, identified by a unique sequence of opcodes.
- **Lifetime:** The number of timesteps in a simulation. The simulation ends either if there are no molecules remaining, or if a pre-specified number of timesteps are executed.
- **Cohort:** the set of molecules in an individual simulation at a given time.
- **Population:** the ensemble of simulations used in a genetic algorithm (note the distinction between 'population' and 'cohort').

Conservation of matter in evolving systems

Real world chemistry conserves matter: in a closed system, the atomic materials are always present throughout the time that the system is closed. Without conservation of matter, there is nothing wrong with a system performing an operation like $X \rightarrow 2X$; the result is potentially unbounded growth. *With* conservation of matter, the reaction can only occur if there are sufficient raw materials in the system to make another instance of X. Thus, growth is bounded.

AChems and artificial evolutionary systems tend to be implemented without the notion of conservation of matter (CoM). As in biological evolutionary systems, artificial evolutionary systems have to have a cohort of entities which exhibit variation in their forms. The variation is set by the initial cohorts and perpetuated through the process of mutation. Selection then operates on the heterogeneous cohort. Unlike in biological evolution, mutation is an event which happens stochastically, and at a pre-specified rate.

Our hypothesis is that conservation of matter will confer the following benefits on an evolving artificial system. Since matter cannot be created or destroyed, the problem of 'bloat' is solved immediately: the system cannot get larger than the resources available to it. In a similar manner, the issue of molecular parasites swamping the system is reduced, since they cannot undergo unbounded exponential growth: resource limits impose a carrying capacity. Conservation of matter could also be linked to mutation and emergence. Limited resources on replication of the genome could lead to errors on copy and temporary limitation of resources may make alternative strategies for survival relatively less costly.

Quantifying the effect by Measuring Evolutionary Activity

We are setting out to test the hypothesis that CoM has a positive influence on the evolutionary activity in a system. One problem with making this link is that the phenomenon of evolvability, also known as evolutionary activity, or the rate of evolution, is vague and difficult to measure quantitatively.

Here we recap the measure of evolutionary activity first presented in Droop and Hickinbotham (2012), which gives a numeric summary of evolutionary activity and thus allows different configurations to be compared.

The reasoning behind the measure is as follows. It is assumed that a species demonstrating some new beneficial adaptation will exhibit a rapidly increasing cohort size. By contrast, a neutrally-drifting species will not show any rapid increase in cohort size. Therefore by creating a measure of cohorts that are rapidly increasing, we can detect evolutionary activity.

Let c_t^i be the number of molecules of species i a timestep t. The total cohort size at timestep t is:

$$C_t = \sum_i c_t^i \tag{1}$$

The proportion of species i at timestep t is:

$$p_t^i = c_t^i / C_t \tag{2}$$

The expected proportion of species i at timestep t is the proportion observed at the previous timestep:

$$e_t^i = \begin{cases} p_{t-1}^i & \text{if } 0 < t \\ 0 & \text{if } t = 0 \end{cases} \tag{3}$$

The activity of species i at timestep t is defined to be the square of the excess of observed over expected proportion, scaled by cohort size:

$$a_t^i = \begin{cases} \left(p_t^i - e_t^i\right)^2 & \text{if} e_t^i < p_t^i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This definition emphasises large positive increases in cohort size for each species, particularly where this occurs within a large cohort.

The total non-neutral activity A_Q of the simulation is the sum of each species activity at each timestep:

$$A_Q = \sum_i \sum_{t=0}^{T} a_t^i \tag{5}$$

The measure A_Q has two advantages. Firstly, it is *quantitative*: it delivers a numerical measure which allows systems to be compared. Secondly, it measures *non-neutral* evolutionary activity without the need for an explicit model of neutral evolution. The measure is called *Quantitative, non-neutral* (QNN) evolutionary activity. Note that QNN is applicable to any systems where changes in cohort over time can be measured. An implementation of the measure in the R programming language is available from the first author's website[1].

Methods

Experimental Vehicle: Stringmol Automata Chemistry

In these experiments, we extended the original configuration of Stringmol described in Hickinbotham et al. (2010, 2012). Stringmol is a modern automata chemistry designed to be much simpler than its forebears by placing less emphasis on registers and memory addressing, and more emphasis on the process of binding as a precursor to a reaction between molecules. For full details of the Stringmol system see Hickinbotham et al. (2012). We give a brief description here. A molecule in Stringmol consists of a string of opcodes and four program pointers. There are no queues for processor time or death; both of these are allocated stochastically.

There are 33 opcodes. Most of these (the alphabetical characters) are 'n-ops': they have no function other than to form sequences of codes which are searched for by the functional codes. The seven functional codes (the non-alphabetical characters) manipulate the position of pointers, using inexact sequence matching to determine program flow.

Molecules run their programs only when they bind to other molecules, as shown in figure 1, meaning that a molecule has no opportunity to interfere with its neighbours, *unless* it can bind to it. This was designed to emulate the specific binding properties of enzymes and substrates, and makes the system much less noisy than other automata chemistries such as Tierra.

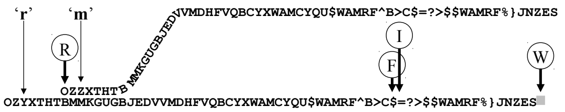

'r'
'm'
R
BMMKGUGBJEDVVMDHFVQBCYXWAMCYQU$WAMRF^B>C$=?>$$WAMRF%}JNZES
I
F
W
OZZXTHT
OZYXTHTBMMKGUGBJEDVVMDHFVQBCYXWAMCYQU$WAMRF^B>C$=?>$$WAMRF%}JNZES

Figure 1: A reaction between two bound molecules 'r' and 'm' in the Stringmol system. Molecule 'm' is being copied by a reaction encoded on molecule 'r'. Pointers are shown as letters in circles, pointing to a position on the molecular sequence.

Molecules survive and multiply by being created more quickly (on average) than they are destroyed. The creation of molecules usually occurs by a process of copying. Figure 1 illustrates a reaction between two molecules after they have bound. The first half of this molecule specifies the binding rate and the second half formulates the program that creates a copy of the molecule. The reaction is initialised by determining which molecule will execute its sequence. The instruction pointer (Ⓘ in figure 1) of the executing molecule 'r' steps through the sequence of opcodes, executing each in turn. It reaches a part of the sequence that executes an iterative copy of the partner molecule 'm'. This is achieved by; copying the opcode at the read pointer Ⓡ to the write pointer Ⓦ; incrementing Ⓡ and Ⓦ; checking the position of Ⓡ, and moving Ⓘ to the flow pointer Ⓕ if the partner molecule is not fully copied. After this has happened, Ⓘ exits the loop, and executes another sequence of instructions that turns the newly-created sequence of opcodes into a new molecule. Cohorts of molecules containing variants of this program that successfully create new copies of each other sufficiently quickly are able to maintain themselves indefinitely.

Conserving Matter in Automata Chemistries

To experiment with conservation of matter, we extend the standard Stringmol framework detailed in Hickinbotham et al. (2012) to conserve matter, to produce CoM-Stringmol. The changes are as follows:

Matter is introduced explicitly by setting a fixed number of instances of each opcode in the system, and keeping a record of how many opcodes of each type are "free", i.e. not forming part of a string molecule. We refer to this record as the *opcode buffer*. This is an array of integers in Stringmol, but it is analogous to the atoms existing in solution in a physical system.

Operators are modified so that it is not possible to change the total number of instances of an opcode. Only the `copy` operator '**=**' needs to be changed in order to achieve this. This operator's function is to insert an opcode at the write pointer (shown as Ⓦ in figure 1), which overwrites any opcode at that location on the string molecule unless the write pointer is at the end of a string. The changes needed in order to conserve matter are straightforward: decrement the

[1] see `http://www-users.cs.york.ac.uk/~sjh/software`

```
WWGEWLHHHRLUEUWJJJRJXUUUDYGRHJLRWWRE$BLUBO^B>C$=?>$$BLUBO%}OYHOB
```

Figure 2: Sequence of the replicase seed molecule used in the experiments

opcode buffer for the opcode indicated by the read pointer (shown as Ⓡ in figure 1), and increment the buffer for any opcode that is overwritten.

Mutation: Occasionally, the `copy` operator will be called when Ⓡ is pointing at an operator that is not available in the opcode buffer. Should this happen, the availability of an alternative (mutant) opcode is checked. If the mutant opcode is available, then it is used, otherwise no opcode is used and the copy operation acts like a deletion. Note that there is no analogy with insertion in this implementation, so the sequences of molecules can get longer only if there is a functional change in the program that the molecule encodes.

In this way, instead of specifying mutation by a simple rate constant as in basic Stringmol, here it is *embodied* as a property of opcode availability, and is dependent both upon opcode concentration and upon the number of opcodes currently used in the sequence of each member of the cohort.

Decay of molecules in Stringmol is instantaneous. The molecule is removed from the simulation with a small probability known as the *decay rate*. In order to conserve matter, the process updates the opcode buffers with the opcodes that return to solution when a molecule is destroyed.

Initialisation: At the start of a run, it is necessary to ensure that the initial cohort of replicas molecules does not comprise more opcodes than are available in the buffer. The initial set of molecules is loaded from a config file and the number of opcodes in the cohort of molecules are counted. If there are too many, then the opcode is deleted from the molecule, and the resulting shortened molecule is retained in the system. There is an argument that this approach introduces variability in the simulation that might distort experiments. We feel that this initialisation approach emulates what might happen if the molecular cohort were created via some earlier simulation where there were no substitutions available, for example when dividing the molecular contents of a cell into two daughter cells.

Seed Molecule

Each run commences with 150 instances of the 'seed' molecule, the sequence of which is shown in figure 2. This is similar to the 'replicase' molecule used in Hickinbotham et al. (2010), but with a more balanced distribution of the n-op code letters (see figure 3). The function of the molecule is identical to the earlier version. In the CoM formulation of Stringmol, replication is exact unless there is a shortage of available opcodes in the opcode buffer.

opcode	count	opcode	count	opcode	count
B	6	**H**	5	**>**	2
C	1	**^**	1	**R**	5
$	4	**J**	5	**=**	1
D	1	**?**	1	**U**	7
E	3	**L**	5	**W**	6
%	1	}	1	**X**	1
G	2	**O**	4	**Y**	2

Figure 3: Number of opcodes used in each of the 150 seed molecules. One seed molecule uses a total of 64 opcodes.

Experiments

We now detail three experiments that characterise the interaction between CoM-Stringmol and evolutionary activity. The first experiment presents a series of evaluations, each with the same fixed concentration for every opcode in the Stringmol chemistry. The second uses an evolutionary search algorithm to find a set of concentrations that produce higher levels of evolutionary activity, by varying the individual concentrations of each opcode. The third experiment examines the effect of low concentrations of opcodes on the amount of evolutionary activity.

Experiment I: Exploring equal concentrations of opcodes

We set the total concentration of each opcode present in the system to a constant λ. We ran a set of configurations varying λ between 1,000 and 3,000 opcodes per container. For example, if we set $\lambda = 1,000$, then since there are 33 opcode types, there would be a total of 33,000 opcodes in the container. For each value of λ, we performed 100 runs of Stringmol using the seed molecule described above. This allows us to benchmark the sensitivity of the QNN measure and evaluate its performance with respect to detecting the effects of CoM.

Our hypothesis is that high opcode concentrations would mean low evolutionary activity since there would be no substitutions or deletions. Low opcode concentrations would mean no functional replicase molecules could be built anew, so the cohort of molecules would be unable to self-maintain, again resulting in little evolutionary activity. Between these extremes, we predict that a "sweet spot" exists, where the concentrations allowed a self-maintaining cohort to exist, but which also induced sufficient opcode substitutions for a higher rate of evolutionary activity to emerge.

We gathered the following statistics for each run: lifetime of the simulation; number of molecular species created; QNN activity. Figure 4 shows the results of these runs.

There are several points to note:

1. The Lifetimes of runs rarely reach the limit of one million timesteps since the population of molecules cannot maintain itself under these conditions. Early extinction is not uncommon in Stringmol without CoM, where the runs would continue indefinitely without mutation. With CoM, runs are particularly short where $\lambda < 2400$.

2. The number of species produced peaks at $\lambda = 1700$. However, the lifetime of runs is very short at the concentration where the number of species is at its maximum. The number of species created could be interpreted as an indicator of evolutionary activity. This exposes a problem with using the number of species as a direct measure of evolutionary activity: systems which generate diversity do not necessarily have the ability to self-maintain. This is the "error catastrophe" of Eigen and Schuster (1977).

3. The QNN as a measure peaks at opcode concentration $\lambda = 2400$. This is the concentration above which the molecular cohort can maintain itself, as indicated by the lifetime of the simulations. The fact that the highest value of the QNN measure is at the boundary between error catastrophe and self-maintaining cohorts indicates that it is a useful measure of evolutionary activity.

In conclusion, this experiment shows that conserving matter in AChem systems not only has an effect on the ability of the system to self-maintain, but also has an ability to affect the rate of evolutionary activity as measured by QNN.

Experiment II: Tuning Opcode Concentration

Apart from expediency, there is no reason why λ should be equal for all the opcodes in a system, just as different minerals are present in different concentrations in the physical world. In this experiment, we use an evolutionary search algorithm to find opcode concentrations that yield high evolutionary activity as measured by QNN.

We hypothesise that evolutionary activity could be improved if the concentrations of different opcodes could be different from each other.

To perform the search, we use the tournament-based microbial GA (Harvey, 2009). The mutation rate was set to 0.1, which is a high value for a genetic algorithm, but it seemed better to have a high value that explored the space of concentrations rather than a low value that might become trapped in a local optimum. There are 33 opcodes in the Stringmol system, so are 33 parameters in the genome encoding the concentration of each opcode. The concentration of each opcode was allowed to vary independently between 0 and 4000, encompassing the range of λ values in Experiment I. Each evaluation required a complete run of Stringmol up to a maximum of 1 million timesteps. Even with a cluster of computers available to us, it was necessary to try to constrain the processing overhead to yield results in reasonable time. Accordingly, we limited the GA population size to 20, and used a single evaluation for each fitness measure, even though this was likely to introduce significant stochasticity into the QNN measure (as apparent from each bar-plot in figure 4). By constraining the processing in this manner, we were able to carry out 5000 tournaments during each of 25 GA runs.

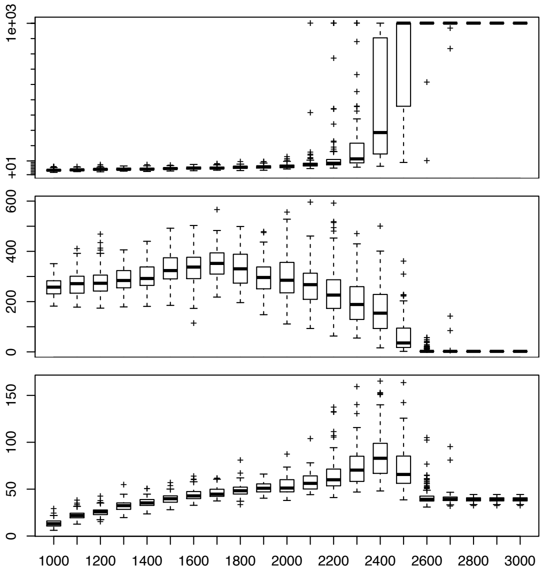


Figure 4: Lifetime (top), number of species (middle) and QNN activity (bottom) for a range of opcode concentrations $1000 \leq \lambda \leq 3000$. 100 runs at each concentration are evaluated.

Results: Fitness changes in GA run We analysed the outputs of the 25 GA runs, and present a summary of the findings here. The left panel of figure 5 illustrates the change in fitness over the 5000 tournaments of three sample runs of the search algorithm. There were two distinct groups: one in which QNN fitness stayed below 200, and one in which a marked increase in fitness occurred at some point in the simulation. 5 of the 25 runs belong to the former group. The principal reason for this distribution can be seen in the right panel of figure 5.

The 20 highest-scoring runs all evolved concentrations of opcodes that allowed the cohort of molecules to self-maintain, as demonstrated by the lifetime reaching the limit set in the simulation. Here, early evaluations in the search went extinct quickly and had with low QNN values, indicating that low concentrations of key opcodes prevent the system from self-maintaining. The evolutionary search then found a set of concentrations that allowed the molecules to self-maintain, and QNN values increased. This is a side effect of using QNN as a fitness measure, as longer simulations have greater opportunity to innovate. In other words,

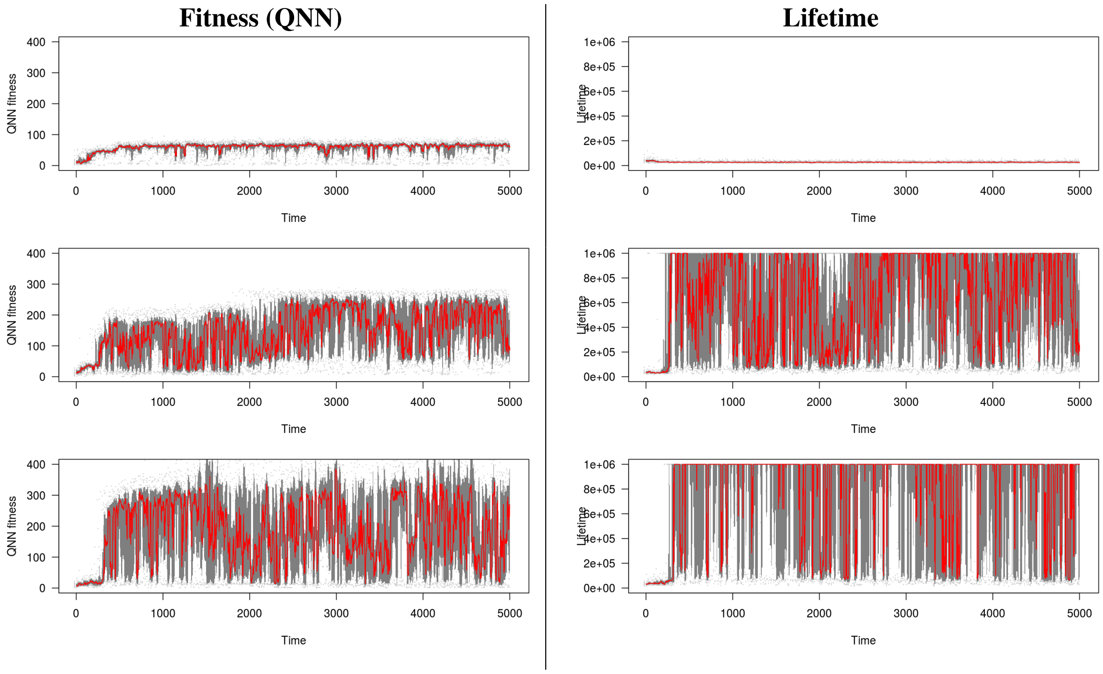


Figure 5: Left: changes in QNN fitness during three selected runs of the microbial GA with 1000 tournaments and a population size of 20. Right: changes in simulation lifetime during these three runs of the search algorithm. In all plots, the red line shows the median value. The shaded area between the blue lines shows the inter-quartile distribution. Points outside the shaded region are outliers.

the QNN-based fitness measure places an indirect requirement on the system to be capable of self-maintaining.

Other important features of the trajectory of the search are:

1. The stochastic nature of individual Stringmol simulations result in variable fitness values for each configuration. However the search algorithm is sufficiently robust to find fit configurations of the opcode concentration levels.

2. Although the median fitness of the simulations increases by several orders of magnitude, individual simulations occasionally score poorly, again due the the stochasticity of the simulation.

3. After the search converges, the median fitness values regularly reach around 300, whereas the median fitness value for a uniform opcode concentration of 2400 was less than 100, leading us to conclude that the search algorithm yielded concentrations of opcodes that improved evolutionary activity.

Results: Opcode Concentrations Figure 6 shows the distribution of opcode concentrations in the final population of the GA runs in figure 5. Although there are no common features in terms of individual opcode concentrations, we find that in each run the concentration of several opcodes is low.

From figure 7 we see that the concentration of the copy operator '=' (lying between the 'T' and 'U' n-ops on the horizontal axis of figure 6) is often (but not always) low. The copy operator '=' has key functionality in the self-replicating system: it specifies that the opcode at the read pointer should be inserted at the location of the write pointer. How can low concentrations of this operator lead to self-maintaining runs with high evolutionary activity? Experiment III addresses this question.

Experiment III: Boosting Concentrations

The previous experiment demonstrates that evolved concentrations of opcodes can yield higher evolutionary activity than setting all opcodes to the same concentration. Some opcodes that are essential for the self maintaining system evolve to low concentrations in high-scoring runs of the GA. Our definition of 'low' concentration is any concentration of less then 400, which is 10% of the maximum permissible concentration in the GA.

The highest scoring run of the 25 we carried out in the previous system had low concentrations of the operators 'D', 'E', '=' and 'X' (see figure 6 and figure 7). Each of these has a different function in the replicase molecule used in our experiments. The 'D' and 'E' are both used to form the part of the sequence that specifies the binding between molecules. The 'X' opcode is also present in the binding region but acts as an indel, serving to *reduce* the chance of molecules bind-

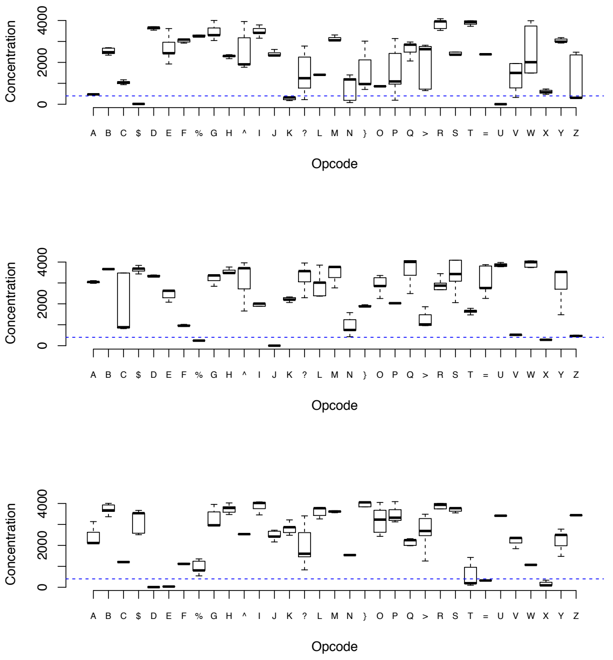


Figure 6: Evolved opcode concentrations for the three GA runs shown in figure 5. The dashed blue line indicates a concentration of 400, which is 10% of the maximum concentration permitted in the GA.

ing. The '=' opcode is central to the copy operation that allows molecules to create copies of other molecules.

We hypothesise that these opcode concentrations need to be low to promote evolutionary activity. To test this, we created a simulation where the four opcodes listed above were present in concentrations at 10 times the evolved value. We call this the 'boosted' system. We then evaluated 20 runs of the evolved system and the boosted system, and measured QNN for each run.

Results Figure 8 shows the distribution of the fittest evolved configuration from experiment I ('fixed'), experiment II ('fittest') and the 'boosted' configuration. We have shown the distributions as boxplots overlaid with beanplots (Kampstra, 2008) to highlight the multi-modal distribution of QNN for the fittest evolved configuration.

Our interpretation of this data is as follows. The 'boosted' configuration is organised into a self-reproducing system that is robust, but demonstrates little evolutionary activity as detected by QNN. By contrast, the fittest configuration sets up dynamics that yield high-scoring evolutionary activity, but those dynamics *also* 'risk' extinction. The configuration evolved because in a *population* of replicating systems, this risk of extinction is ameliorated: there is usually a 'sister' system in the population that is doing well, and preserving the configuration in the population. Both of these configurations demonstrate more evolutionary activity than the best 'fixed' configuration from experiment I, demonstrating that varying the concentrations of opcodes can contribute to the overall innovation of the system.

mean QNN	Scarce opcodes
88.9	`$KU`
124.9	`Z>`
182.1	`>OV`
207.8	`>O G`
213.4	`R`
247.0	`} %J=`
282.2	`%J X`
303.9	`O     %   XD`
332.4	`=   E`
334.9	`=`
352.2	`?ST`
358.0	`O       =`
358.6	`=`
359.1	`=`
361.1	`D`
368.6	`G         ?   M`
369.9	
370.6	`Z   V      =`
375.4	
377.9	`X`
383.0	`D ?`
391.3	`?`
399.5	`V       X   ?`
415.3	`O`
478.8	`=XDE`

Figure 7: Opcodes with low concentrations in 25 runs of the GA, ranked by average fitness at the end of the GA

Discussion

We have demonstrated that conservation of matter can be implemented in an existing ALife system.

In contrast to building a system anew, this approach allows us to explore the pros and cons of conservation of matter, and elucidate the differences between environments where resources are either conserved or available in unlimited quantities. This is important for virtual systems where extra effort is required to ensure that the system conserves matter: we need to establish whether the reward is worth the effort. It is also important for understanding the role of resource availability in physical systems, for example in understanding its role in the origin of life, and in developing self-assembling robotic systems. In our experiments, tuning the concentrations of opcodes to maximise QNN allows us to create CoM-Stringmol runs that innovate and perpetuate, possibly even more than the original Stringmol formulation.

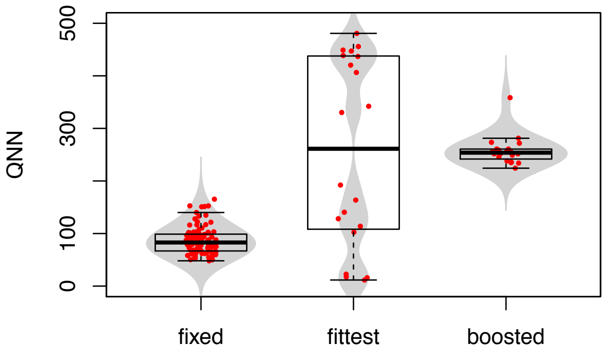


Figure 8: Beanplots (in grey) overlaid with individual data points (in red) and boxplots for distributions of QNN values in three configurations of CoM-Stringmol. **Left:** $\lambda = 2400$ from Experiment I, **Middle:** fittest cohort found in Experiment II, and **Right:** 'boosted' concentrations of scarce opcodes from experiment III.

This contribution is also the first active use of the QNN measure, which was used both to measure a set of values and to drive a search algorithm. As we have seen, the role of the key operators in the Stringmol system can be evaluated at the cohort level, rather than at the level of an individual reaction.

CoM can be thought of as an embodied mutation operator that responds to local conditions. Where the molecular cohort is a good fit to its environment, mutation is low and the system self-maintains. Where the molecular cohort over-exploits resources, mutation increases and new configurations of molecules are found. This is the sort of feature that an evolving system will need in order to better exploit some of the stochastic programming features of Stringmol (see Hickinbotham et al. (2012)), rather than existing in a design space that is adjacent to hand-designed molecules.

Future work: We will attempt to determine what the link is between specific opcode concentrations and high-scoring runs, and how they self-maintain in situations where the concentration of key opcodes is so low. We also intend to apply the concept of CoM to Tierra and Avida, and to richer Stringmol systems.

The QNN measure of evolutionary activity allows us to use conservation of matter to meaningfully explore the local functional molecular space. However, we recognise that the measure should be part of a suite of analysis tools, that should capture information about increasing complexity and diversity in an evolving system.

One of the opportunities this work presents us with is that it allows us to generate different levels of abstraction, and so build more abstract simulations that are correct versions of low level systems like Stringmol. It is important that virtual systems conserve matter if such dual representations are needed (Nellis and Stepney, 2010).

Finally we will use CoM to study the effect chemical flux, to see if systems can adapt to changing ratios of opcodes (Pascal et al., 2013).

Acknowledgements

The experiments were performed using the Volvox cluster at the York Centre for Complex Systems Analysis. Part of this work was funded by the EU FP7 project EvoEvo, grant number 610427.

References

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial chemistries-a review. *Artificial Life*, 7(3):225–275.

Droop, A. and Hickinbotham, S. (2012). A quantitative measure of non-neutral evolutionary activity for systems that exhibit intrinsic fitness. In *ALife XIII*, pages 45–52. MIT Press.

Eigen, M. and Schuster, P. (1977). A principle of natural self-organization. *Naturwissenschaften*, 64(11):541–565.

Fernando, C. and Rowe, J. (2007). Natural selection in chemical evolution. *Journal of Theoretical Biology*, 247(1):152–167.

Harvey, I. (2009). The microbial genetic algorithm. In *Proc ECAL 2009*, volume 5778 of *LNCS*, pages 126–133. Springer.

Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., and Young, P. (2010). Diversity from a monoculture: Effects of mutation-on-copy in a string-based artificial chemistry. In *ALife XII*, pages 24–31. MIT Press.

Hickinbotham, S., Clark, E., Stepney, S., Clarke, T., Nellis, A., Pay, M., and Young, P. (2012). Specification of the stringmol chemical programming language version 0.2. Technical Report YCS-2010-458, Univ. of York.

Hoverd, T. and Stepney, S. (2011). Energy as a driver of diversity in open-ended evolution. In *Proc ECAL 2011*, pages 356–363. MIT Press.

Kampstra, P. (2008). Beanplot: A boxplot alternative for visual comparison of distributions. *Journal of Statistical Software, Code Snippets*, 28:1–9.

Lones, M. A., Fuente, L. A., Turner, A. P., Caves, L. S. D., Stepney, S., Smith, S. L., and Tyrrell, A. M. (2013). Artificial biochemical networks: Evolving dynamical systems to control dynamical systems. *IEEE Transactions on Evolutionary Computation*, pages 145–166.

Nellis, A. and Stepney, S. (2010). Automatically moving between levels in artificial chemistries. In *ALife XII*, pages 269–276. MIT Press.

Pascal, R., Pross, A., and Sutherland, J. D. (2013). Towards an evolutionary theory of the origin of life based on kinetics and thermodynamics. *Open Biology*, 3(11):130156+.

Schneider, E. D. and Sagan, D. (2005). *Into the Cool: energy flow, thermodynamics, and life*. University of Chicago Press.

Evolving Robot Controllers Using Carbon Nanotubes

Maktuba Mohid, Julian F. Miller

University of York, York, UK, YO10 5DD
mm1159@york.ac.uk, julian.miller@york.ac.uk

Abstract

Evolution-in-materio uses computer-controlled evolution to configure materials to solve computational problems. In this paper, the material is a mixture of single-walled carbon nanotubes in an insulating polymer. We show for the first time that using purpose-built hardware it is possible to evolve voltages and signals applied to such materials to control robots, both simulated (Kephera) and real (Pi-Swarm). Evolved controllers were able to fully explore an environment, avoid obstacles, and cope with introduced faults, errors and environment changes. We also evolved a robot controller that could solve mazes of various difficulties.

Introduction

Evolution-in-materio (EIM) is a relatively unexplored area of research which aims to mimic Darwinian evolution by manipulating physical systems using computer controlled evolution (CCE) [Harding and Miller (2009, 2007); Harding et al. (2008); Miller and Downing (2002)]. Like Dawkins "blind watchmaker" metaphor for evolution [Dawkins (1986)], we aim to utilize physical systems without requiring an understanding of their internal processes [Miller et al. (2014)]. We contend that this allows evolutionary algorithms the greatest freedom to find solutions to a given problems. We are trying to avoid Conrad's trap , the so-called "price of programability", which says that in conventional design the vast majority of interactions that could possibly contribute to the problem are deliberately excluded Conrad (1988)! This does not mean that such evolved physical systems cannot be understood. However, it is likely that a complete understanding could only be found after a considerable amount of subsequent analysis.

EIM was inspired by the well-known work of Adrian Thompson who investigated whether it was possible for unconstrained evolution to evolve working electronic circuits using a Field Programmable Gate Array (FPGA). He evolved a circuit that could discriminate between 1kHz or 10kHz signal [Thompson (1998)]. However, Thompson discovered that the evolved circuit operated by exploiting subtle physical properties of the chip. Later Harding and Miller were able to replicate these findings using a liquid crystal display rather than silicon [Harding and Miller (2004)]. In a series of papers they showed that evolved voltages applied to a liquid crystal display could allow it to also implement Boolean logic functions [Harding and Miller (2007)] and act as a simulated robot controller in a simple environment [Harding and Miller (2005)].

We describe the use of a purpose built platform called Mecobo that facilitates computer controlled evolution of a material [Lykkebø et al. (2014)] for controlling a Khepera-like robot and a real Pi-Swarm robot. The Mecobo platform has been developed within an EU funded research project called NASCENCE [Broersma et al. (2012)]. The computational material we have used in this investigation is a mixture of single-walled carbon nanotubes and a polymer. This new platform allows a variety of materials to be investigated in custom designed electrode arrays, using a variety of electrical signals and inputs. One of the aims of NASCENCE is to assess the ability of evolution-in-materio as a methodology for solving a wide variety of computational problems. In recent work, the technique has been applied to solve traveling salesman problems [Clegg et al. (2014)], classification tasks [Mohid et al. (2014c)], function optimization [Mohid et al. (2014b)] and bin-packing [Mohid et al. (2014a)].

Evolutionary computation has been widely used to control robots [Nolfi and Floreano (2001); Bongard (2013)]. EIM has been used before to control simulated robot with wall avoidance behavior. However, the evolvable substrate was liquid crystal [Harding and Miller (2005)]. Here, we give a demonstration that such techniques can be used to control a simulated Khepera robot and a real Pi-Swarm robot [Hilder et al. (2014)] for a desired behavior. In a series of experiments, we aim to evolve a robot controller that allows a simulated robot to explore an enclosed area without colliding with obstacles and to solve mazes with various complexities [Lehman and Stanley (2011)]. At this early stage in this research, we are not claiming that EIM is a competitive method for controlling robots, we are simply trying to demonstrate that evolving configurations of carbon nanotubes has promise and can be used for robot control. This is the first work of its kind. Our experiments provide a yardstick to assess various aspects of EIM using the Mecobo platform. For instance, we can investigate what type of signals are appropriate, and what mixtures of materials give the best results. Using materials in the genotype-phenotype map has, at present, some drawbacks. The main one is the Mecobo platform is relatively slow which means that we can only feasibly evaluate relatively few potential solutions. However, it is a new approach to the solution of computational problems and as the technology is developed it could offer advantages over conventional computational methods [Miller et al. (2014)].

Conceptual Overview

EIM is a hybrid system involving both a physical material and a digital computer. In the physical domain there is a material to which physical signals can be applied or measured. These signals are either input signals, output signals or configuration instructions. A computer controls the application of physical inputs applied to the material, the reading of physical signals from the material and the application to the material of other physical inputs known as physical configurations. A genotype of numerical data is held on the computer and is transformed into configuration instructions. The genotypes are subject to an evolutionary algorithm. Physical output signals are read from the material and converted to output data in the computer. A fitness value is obtained from the output data and supplied as a fitness of a genotype to the evolutionary algorithm [Miller et al. (2014)].

In EIM, a highly indirect genotype-phenotype mapping is employed. An evolutionary algorithm using such a mapping may be able to exploit hitherto unknown physical variables in a material which may increase evolvability. Software-only genotype-phenotype mappings are highly constrained. Banzhaf et al. discussed the importance and potential of physicality and embodiment in [Banzhaf et al. (2006)]. It is not clear what materials are best-suited for EIM. However, a few aspects which appear to be important have been identified [Miller and Downing (2002)]. The material needs to be reconfigurable, i.e., it can be evolved over many configurations to get desired response. A physical material should naturally "reset" itself before applying new input signals on it, otherwise it might preserve some memory and might give fitness scores that are dependent on the past behavior. Preferably the material should be physically configured using small voltage and be manipulable at a molecular level [Miller et al. (2014)].

Mecobo Hardware Platform

The Mecobo hardware platform was designed and built within an EU-funded research project called NASCENCE [Broersma et al. (2012)]. Further details about the Mecobo platform are available in [Lykkebø et al. (2014)]. The material signal interface in Mecobo is very flexible. It not only allows the possibility to evolve which electrode receives an applied signal but also a large variety of configuration signals are available to support materials with different electrical characteristics, from static signals to time dependent digital functions. In the Mecobo platform we have used in this paper (version 3.0), the response from materials can only be sampled as purely static digital signals. Using this platform we can only apply two types of inputs to the material: constant voltage (0V or 3.5V) or a square wave signal. However, different characteristics or input parameters associated with these inputs can be chosen and put under evolutionary control. These input parameters are described in Table 1.

Table 1: Adjustable Mecobo input parameters.

Parameter Name	Description	Note
Amplitude	0 or 1 corresponding to 0V or 3.5V	wave signal amplitude must be 1
Frequency	Frequency of square wave signal	Irrelevant if fixed voltage input
Cycle Time	Percentage of period for which square wave is 1	Irrelevant if fixed voltage input
Start time	Start time of applying voltage to electrodes	Measured in milliseconds
End time	End time of applying voltage to electrodes	Measured in milliseconds

The start time and end time of each input signal determines for how long an input is applied.

Computational Material

The experimental material consists of single-walled carbon nanotubes mixed with polybutyl methacrylate (PBMA) and dissolved in anisole (methoxybenzene). The sample is baked causing the anisole to evaporate. This results in material which is mixture of carbon nanotubes and PBMA. The sample used in our experiments is 1.0% carbon nanotubes by weight (99% by PBMA). Carbon nanotubes are conducting or semi-conducting and role of the PBMA is to introduce insulating regions within the nanotube network, to create non-linear current versus voltage characteristics. The idea is that this might show some interesting computational behavior. Further details of the preparation of experimental material are given in [Mohid et al. (2014c)].

Two electrode arrays are placed in each slide. One sample of experimental material is placed in the middle of each electrode array. Sixteen gold electrodes (eight electrodes on each side) are connected directly with each sample on the electrode array. The electrode array is connected directly with the Mecobo board. The electrode sample is shown in Fig. 1. We have only used one of the electrode arrays in our experiments.

Figure 1: Two electrode arrays, each with carbon nanotubes/polymer sample.

The Robot

The task for the robot is to start from a given position, travel around a closed environment avoiding the walls and obstacles, cover as much floor space as possible and finally, to reach a specific target location. The control system is able to use the distance sensors on the robot, and use this information in the control of the motion of the robot using motors. We have used a simulated Khepera robotic platform for some of the robot experiments. This robot has eight short range infra red sensors and two motors. We have also used a real Pi-Swarm robot in some experiments. This has similar features to the Khepera robot. Generally in evolutionary robotics, evolution is performed in simulation. Solutions based on simulation can be run in faster than using real robot, as it can ignore the physical properties of the robot and its hardware.

EIM Controlled Robot

We adapted a Khepera robot simulator (version 2.0) written by Marcin Pilat [1]. Pilat rewrote a Unix based Khepera written by Olivier Michel [Michel (1996)]. The simulated robot has diameter of 55 nominal units and obstacles or walls are made from small bricks having width and height 20 unit. The map is 1000 X 1000 $unit^2$.

The Khepera simulated robot together with the placement of sensors and motors that have been used in the experiments is shown in Fig. 2. The sensor distance value (in a range [0, 1023], where 0 means no object is found and 1023 means an object is in the nearest position) is calculated as a function of the presence (or the absence) of obstacles (see footnote 1). Random noise corresponding to ±10% of its amplitude is added to the distance value of the sensor. In experiments, the distance values of sensors have been used as inputs to material. The robot moves according to the speed (in a range [-10, 10]) of the motor. Random noise of ±10% is also added to the amplitude of the motor speed while random noise of ±5% is added to the direction resulting from the difference of the speeds of the motors. The motor speed is decided by the output of the carbon nanotube material.

The Pi-swarm robot [Hilder et al. (2014)] is designed as part of the Pi Swarm System, which itself is an extension for the Pololu 3-Pi robot [2], which enables the robot to feature as part of a fully autonomous swarm. The sensors and the two motors are organized in the same way as the Khepera robot (see figure 2). The robot's built in library has a function that can calculate distance between an object and the 8 IR proximity detectors. The distance value has a range [0.0, 100.0] where 100.0 means no object is found. The built in library has functions that accept the motor speed values to drive two motors. The robot's speed is defined to be the range [-1.0, 1.0]) [Hilder (2014)].

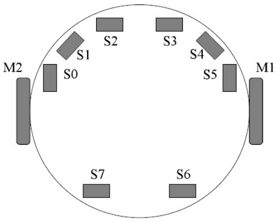


Figure 2: Schematic view of the Khepera and the Pi-Swarm Robot with the positions of the IR proximity sensors (S0-S7) and motors (M1, M2).

Genotype Representation

In case of all experiments, each electrode requires five genes to define how it is used. These decide whether an electrode is used for input or output, or whether it is to receive a configuration voltage. Also, the type of input signal that will be applied to the electrode is decided by evolution. This can be one of the following: signal type, amplitude, frequency, cycle (see table 1).

In some sets of simulated robot experiments only $n_e = 12$ electrodes from the 16 electrodes have been used (the middle 6 electrodes from each side of one material sample). These 12 electrodes were used in the following manner: 6 electrodes were used as inputs, 2 electrodes were used as outputs and remaining 4 electrodes have been used for configuration signals. The 6 inputs were provided by sensors S0, S2, S3, S5, S6 and S7 (see Fig. 2). In other experiments $n_e = 16$ electrodes have been used, where 8 electrodes (all 8 sensors) have been used as inputs, 2 electrodes as outputs and remaining 6 as configuration signals.

Each chromosome consists of $5 * n_e$ genes. The values that genes can take are shown in Table 2 where i takes values 0, 1, $\ldots n_e$. A genotype for an n_e electrode array is shown below:

$$p_0 s_0 a_0 f_0 c_0 \ldots p_{n_e-1} s_{n_e-1} a_{n_e-1} f_{n_e-1} c_{n_e-1}$$

With $n_e = 2m$ (m is 6 or 8 respectively) electrodes, the first $5m$ gene values of a chromosome are related to inputs and they are:
$p_0 s_0 a_0 f_0 c_0 \cdots p_{m-1} s_{m-1} a_{m-1} f_{m-1} c_{m-1}$

and the last 10 gene values of a chromosome are related to outputs and they are:

[1] http://www.pilat.org/khepgpsim/

[2] http://www.pololu.com

Table 2: Description of genotype.

Gene Symbol	**Signal applied to, or read from i^{th} electrode**	**Allowed values**
p_i	Which electrode is used	$0, 1, 2 \ldots n_e - 1$
s_i	Type	0 (constant) or 1(square-wave)
a_i	Amplitude	0 , 1
f_i	Frequency	500 ,501 . . . 10K
c_i	Cycle	$0, 1, \ldots 100$

$$p_{n_e-2}s_{n_e-2}a_{n_e-2}f_{n_e-2}c_{n_e-2}p_{n_e-1}s_{n_e-1}a_{n_e-1}f_{n_e-1}c_{n_e-1}$$

In these input and output genes, only the p_j (here, the values of j are 0-5 and 10-11 for solutions with 12 electrodes and values of j are 0-7 and 14-15 for solutions with 16 electrodes) has any effect, the remainder are redundant. The gene p_j decides which electrode will be used for the inputs and outputs of the device. Thus, mutations in these genes can choose a different electrode to be used as an input or output.

Input Mapping

In all sets of simulated robot experiments, the inputs to the electrode array were square waves with a fixed duty cycle (hereafter referred to as cycle). The cycle time of the square wave signal was determined by a linear mapping of attribute (sensor) data. Denote the i^{th} attribute in a dataset by I_i, where i takes values 0-5 (corresponding to 6 sensors) or 0-7 (corresponding to 8 sensors). Denote the maximum and minimum value taken by this attribute in the whole data set by $I_{i_{max}}$ and $I_{i_{min}}$ respectively. Denote the maximum and minimum allowed cycle times, C_{max} and C_{min} respectively. Then the linear mapping given in Eqn. 1 allows the i^{th} attribute of an instance I_i to map to a square-wave cycle time C_i which was applied to a given electrode.

$$C_i = a_i I_i + b_i \tag{1}$$

where the constants a_i and b_i are found by setting I_i and C_i to their respective maximum and minimum and solving for a_i and b_i.

$$a_i = (C_{max} - C_{min})/(I_{i_{max}} - I_{i_{min}}) \tag{2}$$

and

$$b_i = (C_{min} I_{i_{max}} - C_{max} I_{i_{min}})/(I_{i_{max}} - I_{i_{min}}) \tag{3}$$

In the experiments, $I_{i_{min}}$=0, $I_{i_{max}}$=1023, $C_{min} = 0$ and $C_{max} = 100$. And, input signals all had frequency 5000Hz with amplitude equal to 1.

Equation 1 shows the input mappings used in real robot experiment, but using different input ranges ($I_{i_{min}}$=0 and $I_{i_{max}}$=100) of sensor values. Note also that the distance value of IR detector was subtracted from value 100.0 to maintain similarity with the input mapping used in simulated robot experiment. In simulated robot experiments a lower distance value was used when there was no object was in range of the IR sensor and a higher distance value was used when an object was extremely close to the IR sensor. In the Pi-Swarm experiment, a value 0 was used when no object was found and 100.0 when an object was in the nearest position.

Output Mapping

We determined the output, by examining the output buffers containing samples taken from the output electrodes. Since the Mecobo platform can only recognize binary values, the output buffers contain bitstrings. In all sets of simulated robot experiments, the fraction of ones in the buffer is used to get the output values. This fitness seemed appropriate for inputs that are cycle related. The fraction of ones in the output buffer was linearly mapped to motor speed in the ranges [-10, 10]. The Khepera simulator assumes motor speeds defined in the range [-10, 10]. The mapping is shown in Eqn. 4.

$$o_i = M_{min} + (M_{max} - M_{min}) num_1/num_{total} \tag{4}$$

Where, M_{max} is 10, M_{min} is -10, num_1 is the number of 1's in output buffer and num_{total} is the total number of samples of output buffer and o_i is the output value used to determine the motor speed of the robot. Pi-Swarm experiments also used the fraction of ones and same equation to get output values, but with different output ranges (M_{min}=-1.0 and M_{max}=1.0) of motor speed.

Experiments

Five different sets of experiments were performed with a simulated robot. The task of the robot for the first four sets (set A, B, C and D) was to explore a map without colliding with obstacle. Of them, set C and D used a more complex map, where a number of obstacles were distributed all over the map. In case of experiment D, obstacles were put in the map one by one during the evolutionary run to investigate incremental evolution. Experiment B was designed to investigate fault tolerance under sensor failure [Tyrrell et al. (2004)]. Finally experiment E was concerned with maze solving, where the robot had to solve a number of mazes.

In first set (A), the map shown in image 3 (b) was used. In second set (B), the map shown in image 3 (a) was used. In third (C) and fourth (D) sets of experiments, the map shown in image 3 (c) was used. The fifth set of experiments (E) was concerned with maze solving and six different maps were used in that experiment. Following [Shorten and Nitschke (2014)], the fitness of the robot controller in experiment E was gathered in three stages using three maze maps. The maps in the sequence increased in complexity. The evolutionary run started from the simplest map (shown in image 4 (a)), after an individual in the population was found that could successfully solve the maze, the population was immediately evaluated on the next more complex maze map (shown in image 4 (b)) for further evolution. Once a population member could solve the second maze, the full population was evaluated on the third maze map (shown in image 4 (c)). Once the period of evolution that used the third maze map was completed, the final population of robot controller was tested on three previously unseen other maze maps (shown in image 4 (d), (e) and (f)) to test the generality of the evolved controller. In all of these maps, obstacles are shown in red and the white area of map is the area where the robot is allowed to move.

In the sets of simulated robot experiments A, C, D, only 12 electrodes from the 16 electrodes have been used (the middle 6 electrodes from each side of one material sample). These 12 electrodes were used in the following manner: 6 electrodes were used as inputs, 2 electrodes were used as outputs and remaining 4 electrodes have been used for configuration signals. The 6 inputs were provided by sensors S0, S2, S3, S5, S6 and S7 (see Fig. 2). In experiments B and E all 16 electrodes have been used, where 8 electrodes (all 8 sensors) have been used as inputs, 2 electrodes as outputs and remaining 6 as configuration signals.

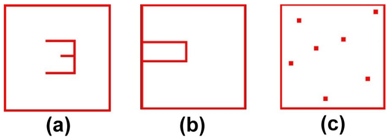


Figure 3: Task environments used simulated robot experiments A-D.

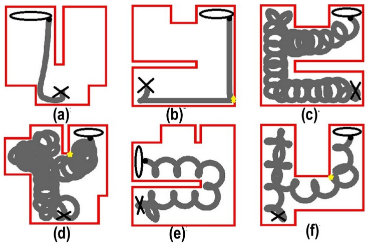


Figure 4: Task environments used in maze solving experiments with examples (according to some experimental results) of paths of evolved robots. In (a)-(f), robot's current position is shown using black ball and the path through which the robot has already visited the map is shown in grey.

For all sets of experiments, each chromosome defined which electrodes were either outputs, inputs (receive square waves) or received the configuration data (square waves or constant voltage).

Using the Mecobo platform the number of samples stored in output buffers can be controlled by the start time, end time and the sampling frequency of output electrode. In all sets of experiments, we used a 25KHz buffer sampling frequency. We applied inputs for a number of milliseconds and accumulated the outputs in a buffer for the same number of milliseconds. We refer to this as input-output timing.

In sets A, C and D of simulated robot experiments, the input-output timing was 20 milliseconds. However, in case of experiments B and E, the input-output timing was 32 and 25 milliseconds respectively due to using a greater number of electrodes. Sampling over longer times is necessary as scheduling in Mecobo is serial. This means that several sequences of actions (i.e., sending input signals, configuration inputs etc) do not take place at the same instant. Mecobo maintains a schedule. Thus, it takes some time for Mecobo to circulate signals to each electrode.

Fitness calculation

To calculate a fitness of an evolved robot controller each individual of the population is executed for a number of time steps. For experiments A-D we used 5000 time steps and for experiments E (maze solving) up to 10,000 time steps. In most cases, if the robot collides with an obstacle, it is stopped immediately resulting in a lower fitness value, however, in case of maze solving experiment, the robot which reaches near to the goal is allowed to run after a collision, for up to 1000 collisions. Also, if a robot rotates in same place for long time, it is also stopped. The latter is assessed using the x and y coordinates of robot's position over 1050 time steps in the past. If the differences between old and new x and y coordinate are ≤ 30 units (approximately half the diameter of the robot), it is assumed that the robot has visited the same place as before, otherwise it is assumed that the robot is exploring a new area of the map. The previous 50 moves of the robot are not used to prevent a slowly moving robot being penalized. Thus, if the robot rotates in approximately the same place for 1000 moves, it is stopped immediately and its overall fitness is assessed. However, in the case of experiments E (maze solving), a robot is not stopped if it rotates in the same place for a long time provided it is able to get close to its goal. If robot has not been stopped early and it is exploring a new area of the map, the distance between previous move and the new move is added to fitness score. The distance is calculated using Euclidean planar distance between the previous and the new move. The better individuals are decided by higher fitness values. But in case of maze solving experiments (E), robots are rewarded for reaching points nearer to the goal. This is done by measuring the Euclidean distance between the position of the robot and the goal. This distance is subtracted from the value $1415 \approx 1414.214$, so that the value becomes higher as robot reaches closer to its goal. Here, it should be noted that the largest distance between any two points in the map is 1414.214 corresponding to the points (0, 0) and (1000, 1000). The obtained value is multiplied by a constant (constant value 10) and then added to the fitness value.

It is quite tricky to decide whether the robot reaches a position close to its goal or not. The Euclidean distance between robot's current position and the goal can be used, but it has a drawback, especially if there is any obstacle between these two positions, in that case, there is no direct path of the robot to reach the goal, thus the robot might need to move a lot to reach the goal. So, merely calculating Euclidean distance between robot's current position and the goal is not a good measure. In the experiment, we have taken this into account by analyzing the positions of all the obstacles. If there is any obstacle between the robot's current position and the goal, the robot is not taken to be near to its goal, otherwise it is decided to be near to its goal.

Transfer of solutions to real robots

Since the Pi-swarm robot has almost same features as Khepera robot, a Khepera simulator can be used as Pi-swarm robot simulator. We tested the final solutions of experiments A and C sets on a real Pi-swarm robot to investigate whether solutions evolved with a simulator perform exactly the same or not. Only successful solutions of simulated robot experiments were used in real robot experiments, i.e. only those solutions were used which explored the full map without colliding with obstacle. The real robot experiments mainly focused on observing whether the robot explored the whole map without colliding with an obstacle. This is why solutions obtained in experiments B and C of simulated robot experiments were not used on real robots as those experiments were designed to test the robot's ability to cope with simulated sensor faults and environmental changes. Also, E set of experiments was concerned with maze solving problem and that was not used in Pi-Swarm experiment as well. The successful solutions from each experiment A and C were tested on a real Pi-Swarm robot.

Simulated Robot Experiments

For each of the experiments a $1 + \lambda - ES$, evolutionary algorithm with $\lambda = 4$ was used. The $1 + \lambda - ES$ evolutionary algorithm has a population size of $1 + \lambda$ and selects the genotype with the best fitness to be the parent of the new population. The remaining members of the population are formed by mutating the parent. In all experiments, mutational offspring were created from a parent genotype by mutating a single gene. Note if there is no offspring that has a higher fitness than the parent, but there is at least one that has a fitness equal to the parent, then an offspring is chosen to be the new parent.

Robot starting positions in experiments varied. In maps (a) and (c) shown in Fig. 3, the starting position was the centre of the map while for map (b), the starting position of the robot was randomly chosen. In the maze solving experiments, the starting position is marked with a cross and the goal is marked with an oval in Fig. 4.

Analysis of Results

In experiments A, ten independent evolutionary runs of 100 generations were carried out. For each individual in the population, the starting position of the robot was randomly selected. In three evolutionary runs, a robot controller was evolved that could explore the full map. Of them, one robot explored the full map and then collided. The other two robots explored the full map without colliding with wall. The minimum number of generation to explore the full map without colliding with wall was 47. It was found from examining the records of robot's movement that the three best exploring robots all started from the upper left corner of the map. This suggests that the starting position of robot plays an important role for the robot for exploring new areas of map.

In experiments B, each of five evolutionary runs of 100 generations were carried out before a fault was introduced. This was done by switching off one sensor (S1) by making the cycle time 0 for the input signal related to that specific sensor. Then each evolutionary algorithm was run for another 100 generations. In all of these cases, the fitness values dropped after adding the fault, however, the robot could recover the fitness value very quickly. Of them, in case of three runs, the fitness value of robot, which was obtained before injecting a fault, was recovered or even exceeded in an average within 42 generations. In one run, the robot could explore the full map before and after adding the fault, but in both of these cases, the robot collided with wall after exploring the map. In two runs, the robot collided with wall without exploring the full map before and after adding the fault. In one run, the robot collided with wall and could not explore the full map before adding the fault, but after adding the fault, it could explore the full map without colliding with wall (in other words the fault was beneficial). In one run, the robot did not collide with a wall, but could not explore the full map before and after injecting the fault.

Experiments C consisted of five evolutionary runs of 300 generations. The third map (c) was used (shown in image 3 (c)). Two robots could explore the full map without colliding with the wall. The other three robots could not explore the full map and also collided with walls. Of them, only one robot could explore 8/9 of the map. The minimum number of generation required to evolve a controller that allowed a robot to explore the whole map without colliding with wall was 198.

The incremental evolution experiments D used 5 independent runs of 300 generations. Other than the 4 side walls, there are 7 obstacles in the middle of the map, which are distributed all over the map. The seven obstacles were added one by one in intervals of 30 generations, starting from the 20th generation. No obstacles were added during the last 100 generations. Whenever new obstacles were added, the fitness dropped, but the robots recovered quickly. However, after 300 generations it was found that none of the five robots could escape collisions with obstacles. Of these, three robots could explore the full map and then collided. Two robots explored 7/9 of the map and then collided with an obstacle. The highest number of moves a robot could survive after evolving for 300 generations was 4261.

Experiments E were concerned with maze solving. Evolution continued until a robot reached the goal. Five independent runs were carried out. Each evolutionary run was performed on three maps incrementally and the final solutions were tested on three other maps. The mazes are shown in Fig 4. The first maze (a) was solved on an average within 20.6 generations, the second was solved on an average in 56.8 generations, the third maze on an average within 6.6 generations.

The final population of each of the five evolutionary runs above was tested on three different mazes to assess generalization. The results are as follows. In first run, maze (d) was solved by 2 individuals of population, maze (e) was solved by none of the five individuals of the population, and maze (f) was solved by three individuals where one individual solved both first and the third mazes; in the second run, the first, second and the third mazes were solved by 2 individuals; in third run, only the first maze was solved and by only one individual; in fourth run, only the second maze was solved by two individuals; in fifth run, the first maze was solved by two individuals, the second maze was not solved by any of the individuals, the third maze was solved by one robot that also solved the first maze. This means, maze (d) was solved by 7 individuals in total in 4 out of 5 runs, maze (e) was solved by 4 individuals in 2 out of 5 runs and maze (f) was solved by 6 individuals in 3 out of 5 runs. It was found that the first maze was solved by more robots than the other two mazes, this is probably due to the fact that it is the simplest maze. Oddly enough, although the third maze was the hardest, it was still solved by 6 robots, which was more than the number of robots that solved the second maze. The examples (according to some results of maze solving experiments) of paths of evolved robots are shown in Figure 4.

Real Robot Experiments

In experiments, the Pi-Swarm robot was run by establishing communication between Pi-Swarm robot (through an mbed microcontroller) and the computer program which communicates with material via Mecobo. The mbed sent distance values from 8 IR proximity detectors on the robot to the computer which sent the two motor speed values back to the robot. The mbed ran the robot using those motor speed values for 10 ms and then stopped the robot and sent 8 distance values to the computer and another cycle began. This sequence of operations were performed 5000 times. Note that the robot was not stopped if it collided with an obstacle unlike the simulated robot. This means that the robot was given chance to free itself if it was stuck. For each move the robot was allowed to run for only 10 ms so that the robot would not move at the time when Mecobo was busy. Prior investigation arrived at the timing of 10 ms. Times larger than 10 ms allowed the robot too much time to move resulting in harder control and more collisions. Using times smaller than 10 ms meant the robot moved very slowly.

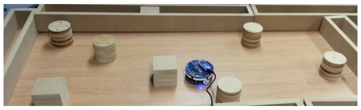


Figure 5: The Pi-Swarm robot moving within a map. The image was taken in the middle of experiment.

The environmental setting of each real robot experiment was the same as the corresponding simulated robot experiment. Only the two successful final solutions of each of two simulated robot experiments (A and C) were tested on the real Pi-Swarm robot. In the case of both sets, one robot explored half of the map after colliding several times with obstacle and then again was able to escape, but the other robot collided very soon after starting it's journey and could not extricate itself within the full life period (within 5000 time steps).

Analysis showed that the real robot did not perform as well as the simulated robot. This was probably due to the presence of noise and also the extra disturbance caused by having a wire connected between the on-board mbed robot controller and the computer while the robot was running. Wireless communication can be possible, but the sent and received messages have limitations on sizes (limited number of bytes). This limitation meant the robot would communicate with computer program too infrequently. Although we attempted to make the settings of simulated and real robot experiments as similar as possible, the organization of map along with obstacles, the proportion of size of robot, the distance traversed by robot with different motor speeds in the map were not exactly the same as those in the simulated robot experiment. Also, the travel time (10 ms) of robot might not be accurate. Figure 5 shows an image of real robot experiment where Pi-Swarm robot is moving within a map.

Conclusions

Evolution-in-materio is hybrid of digital and analogue computing where digital computers are used to configure materials to carry out analogue computation. This is a new concept and has the promise of developing entirely new computational devices. A purpose-built evolutionary platform called Mecobo, has been used to evolve configurations of a physical system to control a robot. The material used is a mixture of single-walled carbon nanotubes and a polymer. In some cases, we found that the robot could explore an environment without colliding with walls. In other recent work using Mecobo we have obtained encouraging results on machine learning classification problems [Mohid et al. (2014c)]. Thus, in principle, a classifier can be implemented using an electrode array and a material sample on a microscope slide and some interfacing electronics. Such a system could act as a low power standalone device. This could have utility in robot control. One of the issues with evolving programs in materials using EIM is that discovering and understanding what is happening inside the material is extremely difficult. One would need very sophisticated measurement and probing techniques which would allow the discovery of important mechanisms. Such probing methods would not have to interfere with the operation of the device. This is a general problem with all evolved physical systems (including biological).

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 317662.

References

Banzhaf, W., Beslon, G., Christensen, S., Foster, J., Képès, F., Lefort, V., Miller, J., Radman, M., and J., R. (2006). Guidelines: From artificial evolution to computational evolution: a research agenda. *Nature Reviews Genetics*, 7:729–735.

Bongard, J. (2013). Evolutionary robotics. *Communications of the ACM*, 56(8):74–85.

Broersma, H., Gomez, F., Miller, J. F., Petty, M., and Tufte, G. (2012). Nascence project: Nanoscale engineering for novel computation using evolution. *International Journal of Unconventional Computing*, 8(4):313–317.

Clegg, K. D., Miller, J. F., Massey, M. K., and Petty, M. C. (2014). Travelling salesman problem solved 'in materio' by evolved carbon nanotube device. In *Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Proceedings*, volume 8672 of *LNCS*, pages 692–701. Springer.

Conrad, M. (1988). The price of programmability. In Herken, R., editor, *The Universal Turing Machine A Half-Century Survey*, pages 285–307. Oxford University Press.

Dawkins, R. (1986). *The Blind Watchmaker*. Norton and Company, Inc.

Harding, S. and Miller, J. F. (2004). Evolution in materio: A tone discriminator in liquid crystal. In *In Proceedings of the Congress on Evolutionary Computation 2004*, volume 2, pages 1800–1807.

Harding, S. and Miller, J. F. (2005). Evolution in materio : A real time robot controller in liquid crystal. In *Proceedings of NASA/DoD Conference on Evolvable Hardware*, pages 229–238.

Harding, S. and Miller, J. F. (2009). Evolution in materio. In Meyers, R. A., editor, *Encyclopedia of Complexity and Systems Science*, pages 3220–3233. Springer.

Harding, S. L. and Miller, J. F. (2007). Evolution in materio: Evolving logic gates in liquid crystal. *International Journal of Unconventional Computing*, 3(4):243–257.

Harding, S. L., Miller, J. F., and Rietman, E. A. (2008). Evolution in materio: Exploiting the physics of materials for computation. *International Journal of Unconventional Computing*, 4(2):155–194.

Hilder, J. (2014). PI Swarm System Manual. http://www.york.ac.uk/media/robots-lab/PiSwarm-v091.pdf.

Hilder, J., Naylor, R., Rizihs, A., Franks, D., and Timmis, J. (2014). The pi swarm: A low-cost platform for swarm robotics research and education. In *Advances in Autonomous Robotics Systems*, volume 8717 of *Lecture Notes in Computer Science*, pages 151–162. Springer.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evol. Comput.*, 19(2):189–223.

Lykkebø, O. R., Tufte, G., Harding, S. L., and Miller, J. F. (2014). Mecobo: A hardware and software platform for in materio evolution. In *Proc. Conf. on Unconventional Computation and Natural Computation*, pages 267–279.

Michel, O. (1996). Khepera simulator version 2.0 user manual.

Miller, J. F. and Downing, K. (2002). Evolution in materio: Looking beyond the silicon box. In *NASA/DOD Conference on Evolvable Hardware*, pages 167–176. IEEE Comp. Soc. Press.

Miller, J. F., Harding, S. L., and Tufte, G. (2014). Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence*, 7:49–67.

Mohid, M., Miller, J. F., Harding, S. L., Tufte, G., Lykkebø, O. R., Massey, M. K., and Petty, M. C. (2014a). Evolution-in-materio: Solving bin packing problems using materials. In *Proceedings of the 2014 IEEE International Conference on Evolvable Systems (ICES): From Biology to Hardware.*, pages 38–45. IEEE Press.

Mohid, M., Miller, J. F., Harding, S. L., Tufte, G., Lykkebø, O. R., Massey, M. K., and Petty, M. C. (2014b). Evolution-in-materio: Solving function optimization problems using materials. In *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pages 1–8. IEEE Press.

Mohid, M., Miller, J. F., Harding, S. L., Tufte, G., Lykkebø, O. R., Massey, M. K., and Petty, M. C. (2014c). Evolution-in-materio: Solving machine learning classification problems using materials. In *Parallel Problem Solving from Nature - PPSN XIII - 13th International Conference, Proceedings*, volume 8672 of *LNCS*, pages 721–730. Springer.

Nolfi, S. and Floreano, D. (2001). *Evolutionary Robotics. The Biology, Intelligence, and Technology of Self-organizing Machines*. MIT Press, Cambridge, MA, USA.

Shorten, D. P. and Nitschke, G. S. (2014). How evolvable is novelty search? In *2014 IEEE International Conference on Evolvable Systems, ICES 2014, Orlando, FL, USA, December 9-12, 2014*, pages 125–132.

Thompson, A. (1998). *Hardware Evolution - Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution*. Springer.

Tyrrell, A., Krohling, R., and Zhou, Y. (2004). Evolutionary algorithm for the promotion of evolvable hardware. *IEE Proceedings Computers and Digital Techniques*, 151(4):267–275.

A Minimal Model for the Emergence of Cooperation in Randomly Growing Networks

Steve Miller[1] and Joshua Knowles[1]

[1]School of Computer Science, University of Manchester, UK
stevemiller.gm@gmail.com

Abstract

Cooperation is observed widely in nature and is thought an essential component of many evolutionary processes, yet the mechanisms by which it arises and persists are still unclear. Among several theories, *network reciprocity* — a model of inhomogeneous social interactions — has been proposed as an enabling mechanism to explain the emergence of cooperation. Existing evolutionary models of this mechanism have tended to focus on highly heterogeneous (scale-free) networks, hence typically assume preferential attachment mechanisms, and consequently the prerequisite that individuals have global network knowledge. Within an evolutionary game theoretic context, using the weak prisoner's dilemma as a metaphor for cooperation, we present a minimal model which describes network growth by chronological random addition of new nodes, combined with regular attrition of less fit members of the population. Specifically our model does not require that agents have access to global information and does not assume scale-free network structure or a preferential attachment mechanism. Further our model supports the emergence of cooperation from initially non-cooperative populations. By reducing dependency on a number of assumptions, this model offers broad applicability and as such may support an explanation of the emergence of cooperation in early evolutionary transitions, where few assumptions can be made.

Introduction

Cooperation is widespread within the natural world and considered to be important in evolutionary processes, particularly in situations where complexity increases, such as early evolutionary transitions, symbiogenesis, or the formation of multicellular organisms (Smith and Szathmary, 1997). A variety of enabling mechanisms have been proposed to explain the emergence and persistence of cooperation (Nowak, 2006), although some of these rely on certain assumptions or constraints being satisfied (e.g. familial relatedness of individuals, or the existence of higher cognitive processes).

Among these enabling mechanisms, *network reciprocity* describes how reciprocal behaviour can be promoted by the form of the connectivity between members of a population. This mechanism appears less demanding in terms of specific assumptions and so potentially offers a more general explanation: most organisms exist within some form of network.

A large body of research has been developed which is focused on understanding network reciprocity. Evolutionary game theory has in particular become a common approach to such investigations, routinely with the use of the single-parameter *weak* prisoner's dilemma (Nowak and May, 1992) as a metaphor for cooperation.

The importance of spatial structure in explaining cooperation was first highlighted in (Nowak and May, 1992). These findings were notably developed with regards to heterogeneous networks in (Santos and Pacheco, 2005, 2006), where it was shown that static heterogeneous networks promote cooperation. Heterogeneity, in network topology, refers to the range of degree values in a network, where degree k represents the number of edges or connections a node may have. In a homogeneously structured network every node has the same value of k. A scale-free (SF) network is considered to have high heterogeneity and has k values distributed according to a power law. Scale-free networks are often assumed to be the result of preferential attachment (PA) processes (also referred to as 'the rich get richer' or the 'Matthew effect' (Merton, 1968)). A key finding in (Santos and Pacheco, 2005) is that scale-free networks (high structural heterogeneity), result in higher levels of cooperation than random networks (low heterogenity). Given that many naturally-occurring networks are considered to have a scale-free structure and hence a power-law degree distribution (Barabási and Albert, 1999; Barabási and Bonabeau, 2003) such findings regarding network heterogeneity and cooperation have naturally generated much interest.

In a particularly interesting work (Poncela et al., 2008), cooperation in dynamic scale-free networks is demonstrated using a coevolutionary model where a preferential attachment (PA) mechanism for network growth is linked to the evolutionary success of evolving agents within a network. A typical PA system involves attachment of newcomers preferentially to those existing individuals which have more network connections. The evolutionary preferential attachment (EPA) model differs from such an approach in that it describes a process where newcomers are more likely to attach to *fitter* members of the existing population. Prior to this work, studies of cooperation in networks had focused on the effect that the network had upon the population. The EPA model specifically adds a causal relationship in the reverse direction, whereby agent behaviour impacts network structure, and hence also offering the possibility of a feedback mechanism.

We recently illustrated that an EPA approach, which additionally incorporates population size fluctuation, supports cooperation, specifically enabling its emergence from *initially non-cooperative* founder networks (Miller and Knowles, 2014). Whilst other models within the literature have investigated pruning of networks by means of link deletion (Zimmermann and Eguíluz, 2005; Santos et al., 2006; Pacheco et al., 2006; Traulsen et al., 2009) or to a lesser extent node deletion (Perc, 2009; Szolnoki et al., 2009; Ichinose et al., 2013), our approach specifically differed from these works in that we deleted nodes on the basis of (least) fitness, thus presenting an evolutionarily representative method of population attrition.

Within this report we extend our previous findings, to develop a minimal model for cooperation in networks which shifts the evolutionary focus from the network growth mechanism to the node deletion process. We demonstrate how such a shift reduces dependency on initial population conditions and also on scale-free network heterogeneity. In the following sections, we first revisit the two key elements of network reciprocity to illustrate how such a shift can still support cooperation; we subsequently expand on the necessity for a minimal model.

Two elements of network reciprocity

Network reciprocity is often explained with reference to heterogeneity. However the promotion of cooperation observed in the foundational work of (Nowak and May, 1992) is due to assortativity of agent strategies (on a homogeneous lattice structure) which results in grouping of cooperators.

Figure 1 presents three identically structured example networks, which feature identical numbers of cooperators and defectors, in order to illustrate how assortativity can promote or suppress cooperation in the *absence of heterogeneity*. These example networks can also be viewed as representative samples of larger networks with homogeneous degree distribution, in which every node has $k = 4$. We extrapolate mean scores (indicated with '$\longrightarrow$') for such larger networks which eliminate the edge effects present in the figures.

Figure 1a, represents a homogeneous (evenly mixed) distribution of strategies, where each node connects to two defectors and two cooperators. Defectors outcompete cooperators in this distribution. Figure 1b shows a non-homogeneous (disassortative) strategy distribution where each agent connects to nonself-similar strategies. In this case the total population score is greater than in Figure 1a, however this strategy distribution results in scores of zero for all cooperators, and positive scores for defectors. Figure 1c illustrates a non-homogeneous (assortative) strategy distribution which shows how self-similar grouping benefits cooperators but not defectors.

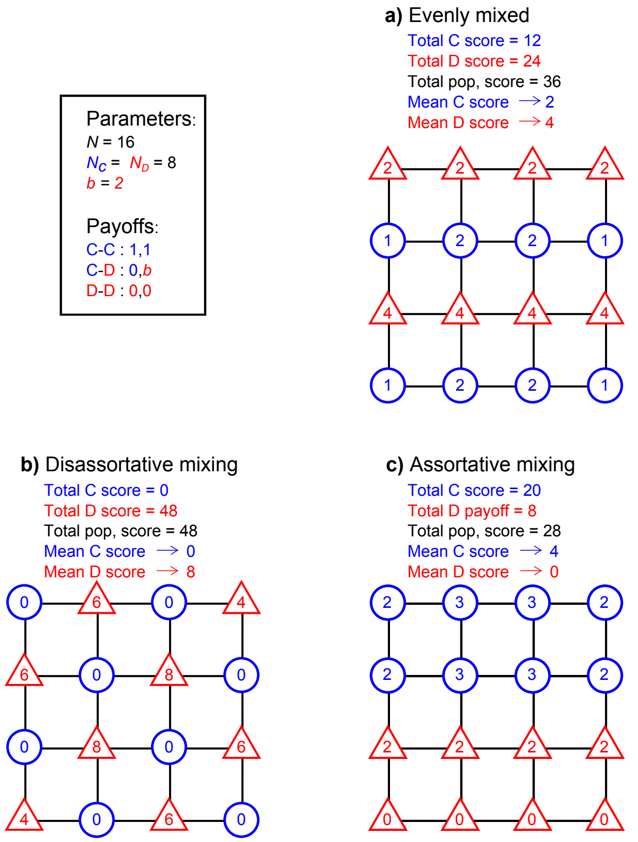


Figure 1: Sample networks illustrating how identical homogeneous network structures can support cooperation to differing extents given differing strategy distributions. Blue circles represent cooperator nodes. Red triangles represent defector nodes. Black lines represent edges (interactions) between nodes. The values within the nodes represent the scores (sum of individual edge payoffs) for the nodes. For illustrative purposes we have selected an arbitrary value of $b = 2$ to calculate example values for the scores.

From these examples, we see that, on a homogeneous spatial structure, given an ability of strategies to redistribute themselves, it is possible for cooperators to improve their lot as a result of self-assorting. Such behaviour allows cooperators to achieve higher scores than would be achieved for random or evenly mixed strategy distributions. From an evolutionary perspective, where scores represent fitness, it is easy to see how self-assortment may allow cooperators to outcompete defectors in a network. It should be noted that defectors do not benefit from self-assorting. (Both strategies benefit from connecting to cooperators and so defectors can therefore only benefit from 'nonself-assorting' behaviour.)

In the work of (Santos et al., 2006) which looks at the effect of heterogeneously (rather than homogeneously) structured networks, the mechanism of assortativity illustrated above supports the formation of groups of cooperators, whilst the greater connectivity that can be found between some individuals in heterogeneous networks offers increased rewards to cooperators in groups. Such models combine the two elements we refer to in this section: *Heterogeneity* further increases the potential gains that can be made by self-*assortativity*.

The volume of scientific literature on the role of heterogeneous networks in cooperation combined with the prevalence of such networks in the real world, tend to leave strategy assortativity as a somewhat marginalised topic. It is worth highlighting that, regardless of the heterogeneity (or lack of) in the network, cooperation cannot emerge without some form of redistribution process that supports grouping of cooperators: Assortativity remains the essential underpinning requirement for spatially structured reciprocity. In this paper we investigate a model for the emergence of cooperation in dynamic networks of evolving agents. Our model makes few demands regarding the specifics of network structure (and associated mechanisms of network formation), whilst still promoting assortativity. We explore the rationale to produce such a model in the following section.

A minimal model

The EPA model (Poncela et al., 2008) introduced earlier has been proposed as a possible explanation for the evolutionary origins of cooperation. The model uses a mechanism for network growth whereby new agents (nodes) added to a network preferentially attach to fitter nodes. The probability that an existing node *i* receives one of the *m* new edges is as follows:

$$\Pi(t) = \frac{1-\epsilon+\epsilon f_i(t)}{\sum_{j=1}^{N(t)}(1-\epsilon+\epsilon f_j(t))}, \qquad (1)$$

where $f_i(t)$ is the fitness of an existing node *i* and $N(t)$ is the number of nodes available to connect to at time *t* in the existing population. The parameter $\epsilon \in [0,1)$ is used to adjust selection pressure. (A fuller explanation of the details of the EPA implementation is provided in the methods section.) Inspection of Equation 1 highlights that in order for a newcomer to "decide" which node to connect to, it is required to have "global" information regarding i) the fitness of all other individuals in the population, and ii) the size of the population, both of which are unlikely to be satisfiable in real world examples.

Given the preferential attachment mechanism, the EPA model is expected to generate a scale-free network (ibid.). Visual assessment of degree distribution supports this hypothesis for certain implementations i.e. for those *b* values where cooperators form the majority strategy.

The scale-free property observed in PA network models parallels many empirical findings in real networks (Barabási and Bonabeau, 2003), however whilst many real networks have been proposed to be scale-free (on the basis of apparent power-law degree distributions), Clauset et al. (2009) have highlighted that such claims are often hypothesised rather than demonstrated. Complex networks are difficult to characterise with certainty (accurately distinguishing power-law distributions from e.g. stretched exponentials is a non-trivial problem) and previous claims of scale-free characteristics in real networks have subsequently been challenged (Amaral et al., 2000; Doyle et al., 2005; Tanaka, 2005). A claim that a network is scale-free is plausible if a preferential attachment process is known to have generated the network, however in the absence of such knowledge, assumptions of scale-free topology may be unreliable. Network models that presuppose scale-free heterogeneity in order to explain cooperation are therefore potentially constrained by such assumptions. We also note that whilst preferential attachment models generate scale-free networks, the converse does not necessarily hold, i.e. while power-law distributions *may* arise as the result of preferential attachment; other approaches can also generate such distributions (Miller, 1957; Albert and Barabási, 2002; Caldarelli et al., 2002).

The application of simplified PA models to real world situations is also impacted by the absence of a general explanation addressing the underlying preferential attachment mechanisms. Each novel situation requires its own explanation. We consider, in particular, the question of how cooperation emerged in early evolutionary transitions. In such situations, involving primitive life forms, which might be for example, immobile, carried by currents and/or interacting randomly, it is unclear whether a mechanism may have existed by which preferential attachment occurred. The ability of fitter individuals to preferentially influence social structure (and hence drive the formation of scale-free networks) cannot be generally assumed.

We now present a minimal model for cooperation in networks which does not require that agents have access to global knowledge, does not depend on a preferential attachment, does not require scale-free network structure, and also does not define or imply feedback between agent behaviour and the population structure.

Our model implements network growth by chronological random attachment (CRA) of new nodes alongside a strategy updating rule which defines agents' evolutionary behaviour. The network is grown by attaching new nodes randomly to existing nodes within the network. We highlight that this does not generate a typical random network (which would have a Poisson degree distribution); instead the chronological nature of the additions results in an exponential degree distribution (Dorogovtsev and Mendes, 2002), with 'older' nodes being more highly connected. The evolutionary strategy updating, which defines strategy assortment, drives the displacement of less fit strategies by those of more successful neighbours. This updating process takes place alongside the growth of the network. For purposes of comparison, our model is based as closely as possible on the EPA model. Strategy updating rules are identical; the node attachment process differs. We also incorporate an additional mechanism not present in the original EPA model: whereas networks established by EPA are fixed in structure once the network reaches a specified size, our model incorporates an attrition process. In this process a certain proportion of less fit members of the population are removed by tournament selection whenever the network reaches a maximum size. The details of EPA and CRA models are presented in the methods section.

Methods

Overview. Our models and simulations are based on those described in (Poncela et al., 2008), but with the addition, in the case of attrition implementations, of a tournament selection step that removes nodes from the network. We here give a full description of the approach for completeness.

The models consist of a network (i.e. graph) with agents situated at the nodes. Edges between nodes represent interactions between agents. Interactions are behaviours between agents playing the one-shot prisoner's dilemma game. These behaviours are encoded by a 'strategy' variable which takes one of two values: cooperate or defect. The game is played in a round robin fashion, with each agent playing its strategy against all its connected neighbours, in turn. Each agent thus accumulates a fitness score which is the sum of all the individual game payoffs.

Within an evolutionary simulation, starting from a founding population, this process is repeated over generations. The evolutionary process assesses agents at each generation on the basis of their fitness score: Fitter agents' strategies remain unchanged; less fit agents are more likely to have strategies replaced by those of fitter neighbours.

The evolutionary preferential attachment (EPA) model connects strategy dynamics to network growth: starting from a small founding population, newcomer nodes are added which preferentially connect to fitter agents within the network. Our chronological random attachment (CRA) model uses the same founding population structures as EPA but adds newcomer nodes to randomly selected existing nodes.

Attrition implementations of both models add a further component which repeatedly prunes the network: Whenever the population reaches a maximum size, a specified percentage of nodes in the network are removed, on the basis of least fitness, after which the network grows again.

Outline of the evolutionary process. Unless stated otherwise in the text, the general outline of the evolutionary process we use is described, for one generation, as follows:

1. *Play prisoner's dilemma*: Each agent plays one-shot prisoner's dilemma with all neighbours and achieves a fitness score that is the sum of all the payoffs.
2. *Update strategies*: Those strategies that achieve low scores are replaced on a probabilistic basis by comparison with the strategies of randomly selected neighbours.
3. *Grow network*: A specified number of new nodes are added to the network, connecting to m distinct existing nodes via *m* edges using either EPA or CRA.
4. *Remove nodes (only in the case of attrition models)*: If the network has reached maximum size, it is pruned by a tournament selection process that removes less fit agents.

In the following, we provide more detail on the specifics of each of the four steps:

Play prisoner's dilemma. We use the single parameter representation of the one-shot prisoner's dilemma as formulated in (Nowak and May, 1992). In this form (the 'weak' prisoner's dilemma), payoff values for the actions, referred to as *T, R, P* and *S*, become b, 1, 0 and 0 (see Figure 2). The b parameter represents the 'temptation to defect' and is set at a value greater than 1 for the dilemma to exist.

From the accumulated prisoner's dilemma interactions, each agent achieves a fitness score as follows:

$$f_i = \sum_{j=1}^{k_i} \pi_{i,j}, \tag{2}$$

where k_i is the number of neighbours that node *i* has, *j* represents a connected neighbour and $\pi_{i,j}$ represents the payoff achieved by node *i* from playing prisoner's dilemma with node *j*.

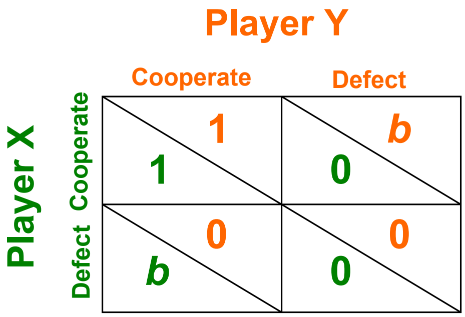


Figure 2: Payoff matrix for weak prisoner's dilemma.

Update strategies. Each node *i* selects a neighbour *j* at random. If the fitness of node *i*, f_i is greater or equal to the neighbour's fitness f_j, then *i*'s strategy is unchanged. If the fitness of node *i*, f_i is less than the neighbour's fitness, f_j, then *i*'s strategy is replaced by a copy of the neighbour *j*'s strategy, according to a probability proportional to the difference between their fitness values. Thus poor scoring nodes have their strategies displaced by the strategies of more successful neighbours.

More precisely, at generation *t*, if $f_i(t) \geq f_j(t)$ then *i*'s strategy remains unchanged. If $f_i(t) < f_j(t)$ then *i*'s strategy is replaced with that of the neighbour *j* with the following probability:

$$P_i = \frac{f_j(t) - f_i(t)}{b.max[k_i(t), k_j(t)]}, \tag{3}$$

where k_i and k_j are degrees of node *i* and its neighbour *j* respectively. The purpose of the denominator is to normalise the difference between the two nodes. The expression $b.max[k_i(t), k_j(t)]$ represents the largest achievable fitness difference between the two nodes given their respective degrees.

Grow network. New nodes, with randomly allocated strategies, are added to achieve a total of 10 at each generation. The probability that an existing node *i* receives one of the *m* new edges was shown in Equation 1. Each new node uses *m* edges to connect to existing nodes. In all our simulations, we use *m* = 2 edges. Duplicate edges and self-edges are not allowed.

Given that in our model each new node extends $m = 2$ new edges, and multiple edges are not allowed, N is therefore determined *without replacement*. The parameter $\epsilon \in [0, 1)$ is used to adjust selection pressure. For all of our simulations $\epsilon = 0.99$, hence focusing our model on selection occurring directly as a result of the preferential attachment process.

In CRA implementations, new nodes connect to existing nodes randomly. The probability that an existing node *i* receives one of the *m* new edges becomes simply:

$$\Pi(t) = \frac{1}{N(t)}. \tag{4}$$

Remove nodes (in the case of attrition models). On achieving a specified size, the network is pruned by a percentage *X*. This is achieved by tournament selection using a tournament size equivalent to 1% of the population. The tournament members are selected randomly from the population. The tournament member having the least fitness is the 'winner'. The remaining nodes are returned to the population. By this method, a shortlist of $X\%$ nodes is established for removal from the network. All edges from deleted nodes are removed from the network. Any nodes that become disconnected from the network as a result of this process are also deleted. (Failure to do this would result in small numbers of single, disconnected, non-playing nodes, having static strategies, whose zero fitness values would result in continual isolation from the network.) When there are multiple nodes of equivalent low fitness value, the selection is effectively random (on the basis that the members were originally picked from the population randomly). Where $X = 0$, no attrition occurs.

General simulation conditions. We investigated networks grown from an initial complete network with $N_0 = 3$ agents at generation t_0. Founding populations were either entirely cooperators or entirely defectors. Networks were grown to a maximum size of $N = 1000$ nodes with an overall average degree of approximately $k = 4$. Simulations were run until 2000 generations. The 'fraction of cooperators' values we use (denoted by $\langle c \rangle$) are means, averaged over the last 20 generations of each simulation, in order to compensate for variability that might occur if just using final generation values. Each simulation consisted of 10 replicates. We used $X = 2.5\%$ for all simulations.

Results

We compared the effects of non-attrition vs. attrition ('+') implementations of two models: i) chronological random attachment (CRA and CRA+), and ii) evolutionary preferential attachment (EPA and EPA+).

How do preferential and random attachment affect the models? In Figure 3 we show profiles of final levels of cooperation, $\langle c \rangle$ vs. temptation to defect, *b* for the four implementations. We note that whilst the preferential attachment mechanism appeared (on the basis of log-log linearity) to result in scale-free networks for those values of *b* where cooperation is supported, it is hard in the light of these results to argue that such scale-free networks necessarily achieve higher levels of cooperation than those formed by random node addition (which have exponential degree distributions): Examining non-attrition implementations of the models (solid lines), we see that in the case of networks grown from cooperative founders, EPA results in higher levels of cooperation than CRA. However, the same cannot be said for populations grown from defectors where random attachment results in higher levels of cooperation for values of $b < 1.3$. Why does random attachment benefit cooperation in defector-founded networks?

We attempt to answer this using Figure 4 where we illustrate early node attachment to cooperator- and defector-founded populations. In Figure 4a, the three cooperator founders have their scores reinforced by their interconnections (shown with bold lines). Added nodes (dashed lines) differ in their scores by a factor of *b* depending on whether they are cooperator or defector. Nodes added to a cooperator-founded network will initially tend to have scores ($= m$ or $m * b$) *relatively* similar to the founder nodes (= 2 or 3), with precise values dependent on *b*. Figure 4b illustrates that self-similarity within the founder network does not benefit defectors in the same way that is seen for cooperators (D-D interactions results in payoffs of zero for both individuals). Whilst this seems to be a weakness, defector-founded networks have an alternative advantage: the addition of new nodes which are cooperators increases the founders' scores, whilst the newly added cooperators score

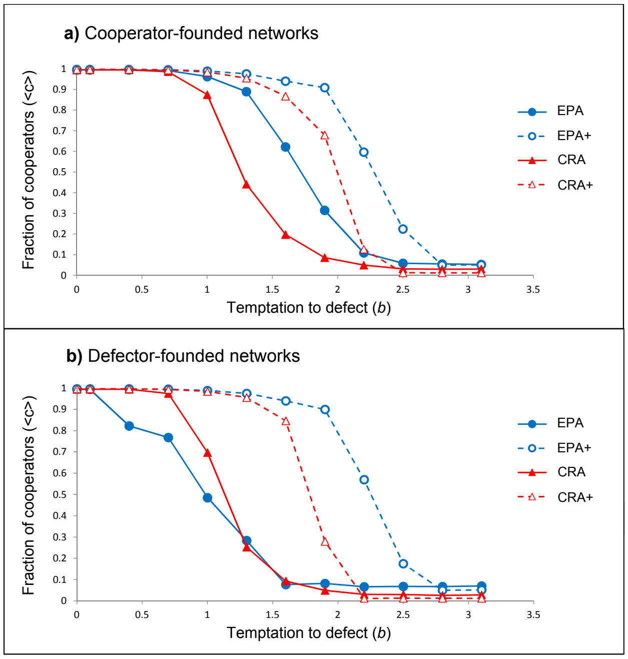


Figure 3: The effect of network model on the relationship between temptation to defect and cooperation. Each line is the average of 10 replicate simulations. Simulations featured 1000 nodes and were run for 2000 generations. Non-attrition models achieved fixed network structure after 100 generations beyond which strategy updating continued alone. Attrition implementations (EPA+ & CRA+) are represented with dashed lines.

zero. Added defectors score zero and only add to the mass of defectors present. In summary, nodes initially added to a defector-founded network score zero, regardless of strategy.

The interactions within these networks are complex and subject to random events. Outcomes are not assured, however we see that *for defector-founded networks*, relative score differences will create a high scoring founder network with newcomers unable to score. Whereas in a cooperator-founded network, scores will be relatively similar between newcomers and existing nodes, in defector-founded networks, we see a disparity in scores which causes an initial bias against cooperation. Cooperators in this situation are likely to be converted to defectors. Preferential attachment, by definition, drives new nodes to connect to the higher scoring founder members. It therefore promotes the disparity in scores and reduces the likelihood of new nodes attaching to non-founders, although it does not eliminate the possibility. (The bias against cooperation that we have described can be overcome in the less probable situation where new cooperator nodes connect to other cooperator nodes rather than the initial defector founders.)

We now refocus on the initial question of why CRA is able to support cooperation from defector-founded networks to a greater extent than EPA. In CRA, by definition, new nodes are connected randomly. Unlike EPA, high scoring defector-founder networks have no enhanced ability to preferentially attract newly added cooperators (and then convert them to defectors). Without this 'pull' of new node connections to the defector founders, alternative interconnected groups can form, which would be more likely to support cooperation.

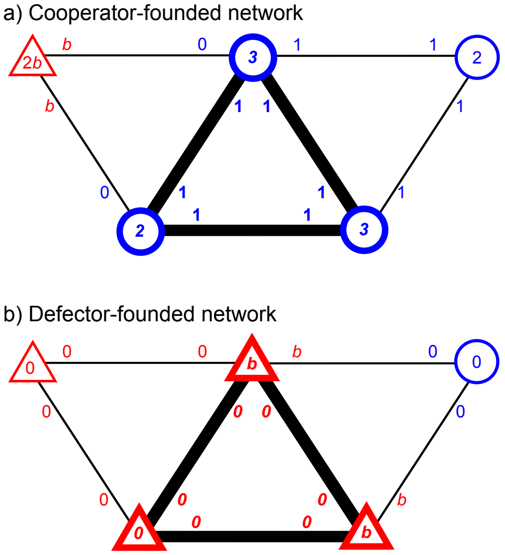


Figure 4: Initial interactions between founder networks and added nodes for **a)** cooperator- and **b)** defector-founded networks. Blue circles represent cooperators. Red triangles represent defectors. Interaction payoffs are shown along edges. Cumulative scores are shown within nodes. Founder networks are shown in bold.

We note, in terms of asymptotic outcomes for CRA networks, that given random attachment of new nodes, the greater the number of nodes in the network the smaller is the influence of the founders. This situation is clearly very different to EPA where early nodes are likely to develop into influential hubs. Specifically for EPA simulations, in the case where defectors found a population, we see the interesting result that the "rich get richer" effect is detrimental to the interests of cooperators.

How does attrition change outcomes for these models? We see from Figure 3 that when the network models incorporate repeated attrition of least fit members (see dashed lines) cooperation is promoted, regardless of founder population strategy type. Attrition increases cooperation for both network formation models and reduces dependence on initial conditions. In direct comparisons, EPA with attrition achieves higher levels of cooperation than CRA with attrition.

In the case of EPA, we know that the network structure becomes fixed (by generation 100 in our simulations), and any increase in cooperation thereafter is due to strategy updating. When attrition is added to the model, cooperation increases, implying that the early fixation of network structure limits achievable levels of cooperation. The attrition model effectively removes this limit and allows the network to continually restructure simultaneously with strategy redistribution. Specifically, it has been postulated in (Miller and Knowles, 2014) that random events during the earliest stages of a network's formation can have long-term consequences for cooperation. We have illustrated one example of how such effects may arise in Figure 4. Given structural fixation, such consequences are 'locked-in'. Attrition allows for some opportunity to 'unlock' the structure.

More generally, for both CRA and EPA models, attrition targets low fitness nodes. We can estimate some information about the type of strategies and the connectivity of such low fitness nodes. They are likely to be: i) defectors amidst a 'sea' of other defector nodes (D-D payoffs are 0, 0), or ii) marginalised defectors with degree $k = 1$ ('terminal nodes'), connected to one other defector. We note that low degree cooperators connected to defectors are unlikely to be evolutionarily stable: their strategies would be displaced by defection.

It seems that attrition, by focusing on least fit members of the population, eliminates defector occupied nodes. Replacement strategies however will then be either cooperators or defectors with equal probability, and the nodes they occupy may also reattach to more opportune positions in the network.

Does attrition change the topology of the CRA networks? We show in Figure 5 that CRA+attrition does not result in a different type of degree distribution to CRA: there is no increase in the range of degree values. If anything, for very high values of b, the opposite is true: heterogeneity of degree is reduced for the attrition networks. The reduced frequency of higher degree nodes is explained by the fact that for high values of b the simulation will be entirely overrun by defectors so most agents will achieve scores of zero, regardless of degree. Given the uniformity of fitness values presented by this scenario, attrition becomes a random process rather than specifically being biased towards low fitness. Note that the presence of nodes of degree $k = 1$ in the Figure is an artefact caused by the attrition process (deletion of some nodes occasionally leaves other residual nodes of degree $k = 1$ in the network).

Conclusions

We have demonstrated a model, featuring network growth by a random process, which supports network-reciprocal cooperation. The key to cooperation in our model is the mechanism of assortativity which allows agents in the simulation to capitalise on the exponential degree distribution of the network such that cooperators can form groups which serve to elevate the rewards available to them. Assortativity is promoted by the evolutionary attrition process included within our model whereby less fit nodes are deleted.

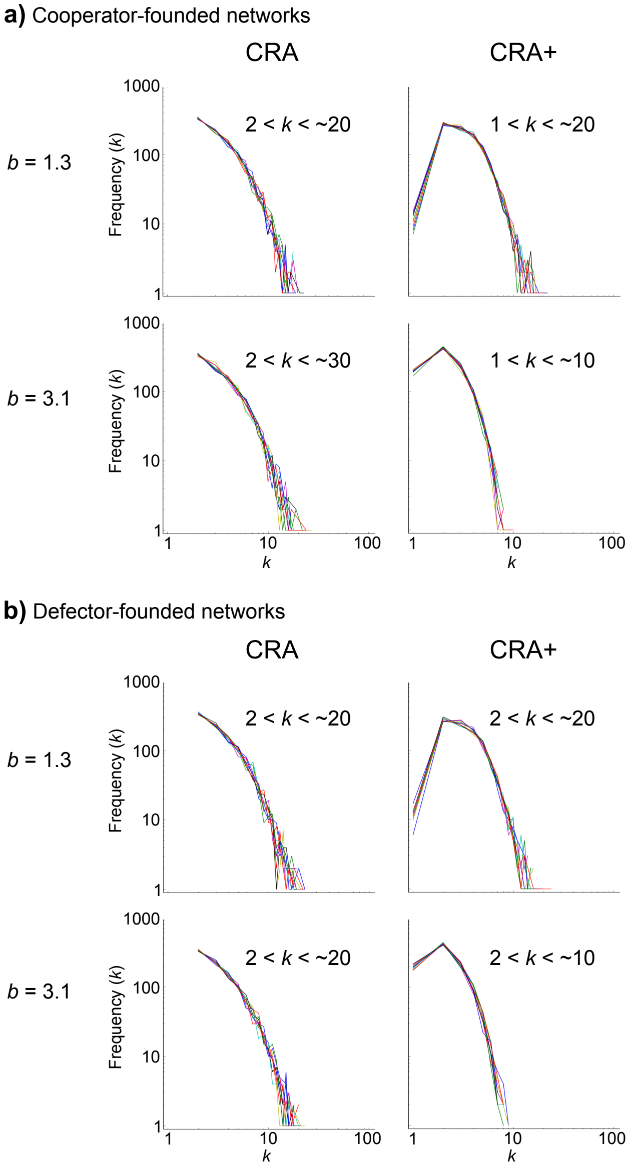


Figure 5: Degree distributions for networks formed using CRA vs. CRA+attrition for low and high temptation to defect (b) values. **a**) shows results for networks grown from cooperator founders and **b**) for defector founders. Observed range of degree values is shown within each plot.

We find that our model of chronological random attachment with fitness-based attrition (CRA+), supports levels of cooperation equivalent to or greater than an existing coevolutionary method based on preferential attachment (EPA). Our model does not require that agents have memory or higher cognitive abilities and it supports cooperation, regardless of the behaviours initially present amongst the founding members of the population. Importantly, the model supports cooperation without the requirement for agents to have any form of global knowledge regarding either the network or other members of the population. Given random linking of new nodes, there is also no requirement to explain a mechanism for preferential attachment.

Our minimal model points to a possible general explanation applicable to the emergence of cooperation in networks of primitive individuals. The requirements are that new nodes are added over time to an existing network, and that cooperative behaviours which increase the fitness of individuals have a tendency to persist over less beneficial behaviours — the latter being eliminated by evolutionary selection.

We have found in comparing these models of network growth that network-reciprocal cooperation can exist without the level of degree heterogeneity associated with scale-free structure. Such findings at first glance appear somewhat at odds with the prevailing consensus that increasing heterogeneity promotes cooperation; however by explicitly considering the benefits to cooperation offered by the combined effects of heterogeneity and assortativity, we can shed a different light on our results. Whilst our model clearly introduces only a very limited form of heterogeneity in terms of the network structure, our findings allow for the possibility that, with regards to cooperation, it is more beneficial that cooperators can maximise a non-homogeneous self-assortative strategy distribution, than the network structure itself be highly heterogeneous.

Acknowledgements

This work has been funded by the Engineering and Physical Sciences Research Council.

References

Albert, R. and Barabási, A.-L. (2002). Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47.

Amaral, L. A. N., Scala, A., Barthelemy, M., and Stanley, H. E. (2000). Classes of small-world networks. *Proceedings of the national academy of sciences*, 97(21):11149–11152.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Barabási, A.-L. L. and Bonabeau, E. (2003). Scale-free networks. *Scientific American*, 288(5):60–69.

Caldarelli, G., Capocci, A., De Los Rios, P., and Muñoz, M. A. (2002). Scale-free networks from varying vertex intrinsic fitness. *Physical review letters*, 89(25):258702.

Clauset, A., Shalizi, C. R., and Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, 51(4):661–703.

Dorogovtsev, S. N. and Mendes, J. F. (2002). Evolution of networks. *Advances in physics*, 51(4):1079–1187.

Doyle, J. C., Alderson, D. L., Li, L., Low, S., Roughan, M., Shalunov, S., Tanaka, R., and Willinger, W. (2005). The "robust yet fragile" nature of the Internet. *Proceedings of the National Academy of Sciences of the United States of America*, 102(41):14497–14502.

Ichinose, G., Tenguishi, Y., and Tanizawa, T. (2013). Robustness of cooperation on scale-free networks under continuous topological change. *Physical Review E*, 88(5):052808.

Merton, R. K. (1968). The Matthew effect in science. *Science*, 159(3810):56–63.

Miller, G. A. (1957). Some effects of intermittent silence. *The American Journal of Psychology*, 70(2):311–314.

Miller, S. and Knowles, J. (2014). Population Fluctuation Promotes Cooperation in Networks. *arXiv preprint arXiv:1407.8032*.

Nowak, M. A. (2006). Five rules for the evolution of cooperation. *Science*, 314(5805):1560–1563.

Nowak, M. A. and May, R. M. (1992). Evolutionary games and spatial chaos. *Nature*, 359(6398):826–829.

Pacheco, J. M., Traulsen, A., and Nowak, M. A. (2006). Active linking in evolutionary games. *Journal of theoretical biology*, 243(3):437–443.

Perc, M. (2009). Evolution of cooperation on scale-free networks subject to error and attack. *New Journal of Physics*, 11(3):033027.

Poncela, J., Gómez-Gardeñes, J., Floría, L. M., Sánchez, A., and Moreno, Y. (2008). Complex cooperative networks from evolutionary preferential attachment. *PLoS one*, 3(6):e2449.

Santos, F. C. and Pacheco, J. M. (2005). Scale-free networks provide a unifying framework for the emergence of cooperation. *Physical Review Letters*, 95(9):098104.

Santos, F. C. and Pacheco, J. M. (2006). A new route to the evolution of cooperation. *Journal of Evolutionary Biology*, 19(3):726–733.

Santos, F. C., Pacheco, J. M., and Lenaerts, T. (2006). Cooperation prevails when individuals adjust their social ties. *PLoS Computational Biology*, 2(10):e140.

Smith, J. M. and Szathmary, E. (1997). *The Major Transitions in Evolution*. Oxford University Press.

Szolnoki, A., Perc, M., Szabó, G., and Stark, H. (2009). Impact of aging on the evolution of cooperation in the spatial prisoner's dilemma game. *Physical Review E*, 80(2):021901.

Tanaka, R. (2005). Scale-rich metabolic networks. *Physical review letters*, 94(16):168101.

Traulsen, A., Santos, F. C., and Pacheco, J. M. (2009). Evolutionary games in self-organizing populations. In *Adaptive Networks*, pages 253–267. Springer.

Zimmermann, M. G. and Eguíluz, V. M. (2005). Cooperation, social networks, and the emergence of leadership in a prisoner's dilemma with adaptive local interactions. *Physical Review E*, 72(5):056118.

The Kinetic Basis of Morphogenesis

Yuri Shalygo[1]

[1]Gamma Ltd, Vyborg, Russia
yuri.shalygo@gmail.com

Abstract

It has been shown recently (Shalygo, 2014) that stationary and dynamic patterns can arise in the proposed one-component model of the analog (continuous state) kinetic automaton, or kinon for short, defined as a reflexive dynamical system with active transport. This paper presents extensions of the model, which increase further its complexity and tunability, and shows that the extended kinon model can produce spatio-temporal patterns pertaining not only to pattern formation but also to morphogenesis in real physical and biological systems. The possible applicability of the model to morphogenetic engineering and swarm robotics is also discussed.

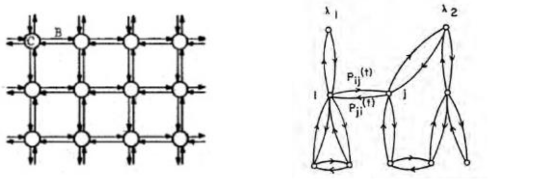
Introduction

In his seminal paper on morphogenesis (Turing, 1952), Alan Turing demonstrated that different spatio-temporal patterns can arise due to instability of the homogeneous state in reaction-diffusion systems. It has been shown recently (Shalygo, 2014) that stationary and dynamic patterns can also arise in the proposed one-component model of the analog (continuous state) kinetic automaton, or kinon for short, defined as a reflexive dynamical system with active transport. This paper presents extensions of this model, increasing further its complexity and tunability. The main aim of this paper is to show that anisotropic diffusion, usually regarded as anomalous, in fact is quite ubiquitous and can be harnessed in morphogenetic engineering and robotics.

The proposed model stems from a number of existing models of complex dynamical systems, and the following in particular:

- *Cellular Automata* (CA) conceived in 1950's by John von Neumann and Stanislaw Ulam.
- *Coupled Map Lattices* (CML) proposed in 1985 by Kunihiko Kaneko as a paradigm for the study of spatio-temporal complexity.
- *Lattice Gas Automata* (LGA) introduced in 1986 by Frisch et al and Stephen Wolfram independently for the modeling of fluid dynamics.
- *Lattice Boltzmann Model* (LBM) evolved from LGA and attracting growing popularity in Computational Fluid Dynamics (CFD) and other fields (Chopard et al, 2002).

Nevertheless, a decisive impetus for the kinon model was given by Konrad Zuse's *net automaton* (Zuse, 1969) and Gordon Pask's *diffusion network* (Pask, 1961) [Fig.1].



a) Zuse's Net Automaton *b)* Pask's Diffusion Network

Fig.1 *Dynamic systems with active transport*

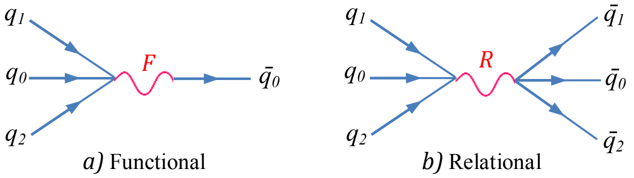
The central idea behind these networks is that nodes of the network are connected reciprocally with the lines that have not only transport but also storage functions. This is in sharp contrast to the conventional view on network links as passive elements. None of the existing models can be applied to Pask's diffusion networks; therefore, a new generation of topology and state space invariant models with active links is needed. The basic kinon model, introduced in the previous paper and outlined in the next paragraph, is a trial step in this direction.

Background

The majority of the existing models are discrete time networks, in which values assigned to nodes and representing their current state are updated synchronously by some transformation. According to the type of transformation, they can be divided in two main groups: *functional* and *relational*.

Functional transformation maps a set of input values onto a single (scalar) output value - a new state of the node, which is relayed or fanned out to all output links. Formally, functional transformation is a many-to-one map $F\colon Q^{k+1} \rightarrow Q$, where Q is a set of states of the node and its k neighbors [Fig.2a].

Relational transformation maps a set of input values onto a set (vector) of output values of the same dimension, therefore it is homomorphism or a structure preserving (one-to-one) map of the form $R\colon Q^{k+1} \rightarrow Q^{k+1}$ [Fig.2b].



a) Functional *b)* Relational

Fig.2 *Feynman diagrams of state transformation*

The difference stems from different model structures. In functional models, the state of a cell is represented by a scalar value q_0 [Fig.3a]. In relational models, it is a vector $\{q_0,q_1...q_k\}$ associating the first component with a cell and the other ones with its k neighbors [Fig.3b]. Contrary to functional models, the value q_0 is not observable to the neighbors of the cell. The values $q_1...q_k$ represent the feedback (observables) of the cell and reside in the links responsible both for information propagation and storage. It implies the *dualism* of relational models, reincarnating as *autonomous cells* during collision or *autonomous links* during propagation.

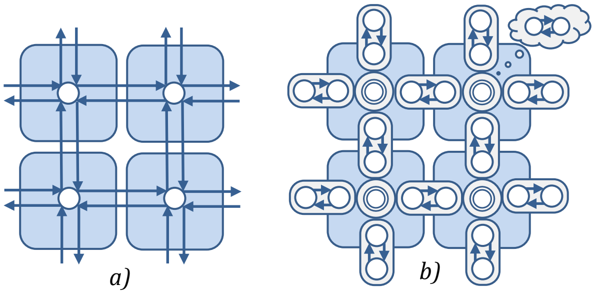


Fig.3 *Structure of functional (a) and relational (b) models*

Similar to LGA and LBM, the kinon model is *relational* and *quantity conservative*, because it was designed to be able to simulate real physical phenomena. CA, CML, Random Boolean Networks, Artificial Neural Networks, etc. are functional and non-conservative in general.

The kinetic automaton can be viewed as the generalization of LBM, which is not restricted to the Boltzmann equation and a regular grid. The key element of the model, making its properties and dynamics different from LBM, is a collision step which was transformed into a 3-step operator (Encoding-Modulation-Decoding) called Conservative Rank Transform (CRT). In this method, not quantities as such but their relative values (ranks) are transformed (modulated), and the total quantity does not change after transformation.

The model represented in Fig.4 was elaborated with having in mind Rosen's Modeling Relation (Rosen, 1991) as well as Kauffman's autonomous agent doing its own work-constraint cycle (Kauffman, 2000).

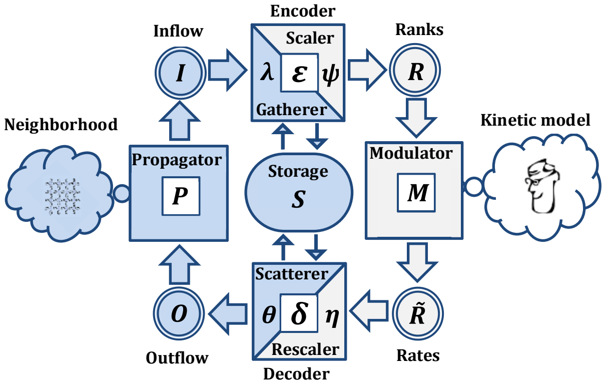


Fig.4 *Kinon State-Transition Structure*

Formally, a kinon is a 9-tuple ($I,R,\tilde{R},O,S,P,\varepsilon,M,\delta$),where:

I - vector of absolute $\mathfrak{R}^+$ input values ($I_1 ... I_k$) *(inflow)*,
R - vector of relative *[0,1]* input values (R_0, $R_1...R_k$) *(ranks)*,
$\tilde{R}$ - vector of relative *[0,1]* output values ($\tilde{R}_o$, $\tilde{R}_1$... $\tilde{R}_k$) *(rates)*,
O - vector of absolute $\mathfrak{R}^+$ output values ($O_1 ...O_k$) *(outflow)*,
S - two-vector of absolute $\mathfrak{R}^+$ values (S_i, S_o) *(storage)*,
P - propagation operator P: $O \rightarrow I$ *(propagator)*,
ε - encoding operator ε: (I, S_o) $\rightarrow$ (R, S_i) *(encoder)*,
M - modulation operator M: $R \rightarrow \tilde{R}$ *(modulator)*,
δ - decoding operator δ: ($\tilde{R}$, S_i) $\rightarrow$ (O, S_o) *(decoder)*.

Encoding (ε) is a composition of gathering (λ) and scaling (ψ) operators: $\varepsilon = \psi \circ \lambda$. Similarly, decoding ($\delta$) is composed of rescaling (η) and scattering (θ) operators: $\delta = \theta \circ \eta$.

Schematically, the kinon internal structure can be represented in more detail by the diagram in Fig.5:

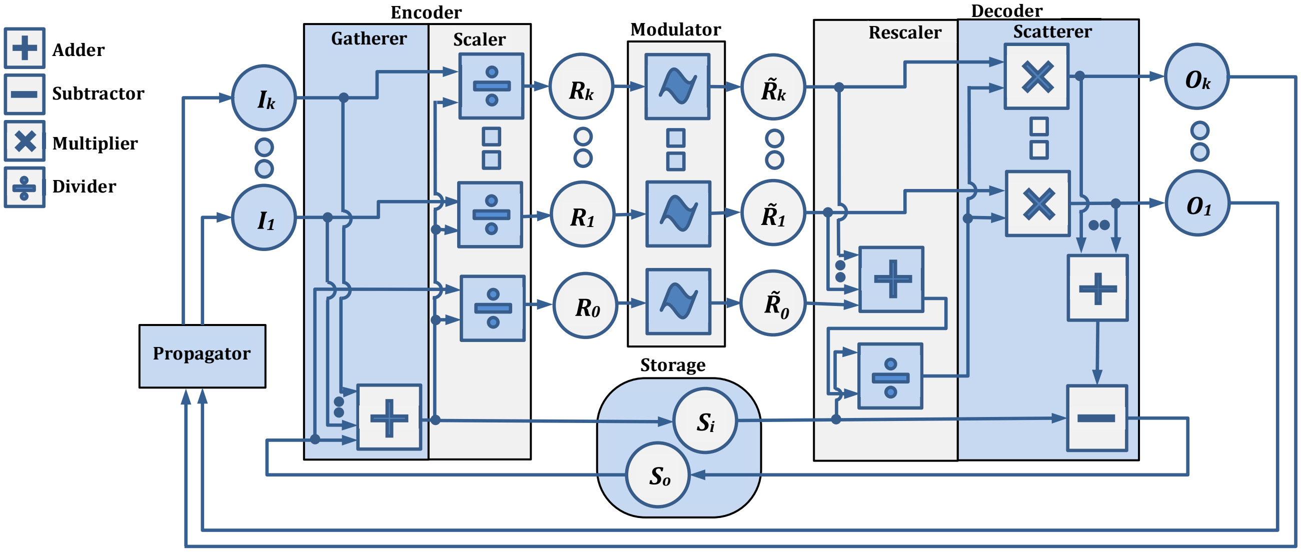
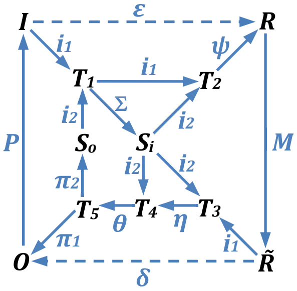


Fig.5 *Schematic diagram of the basic kinon model*

Since all operators are relational transformations or morphisms, the kinon model is very congenial to category theory and categorical system theory (Louie, 1983) in particular; thus the categorical meta-language and notation fit the structure and functioning of kinetic automata quite naturally. The categorical diagram in Fig.6 gives a compact representation of *the algorithm* of the basic kinon model and clearly shows the usage of storage in encoding and decoding and a pivotal role of the input storage *Si* during these steps.



$$T_1 = P\,(O) \oplus S_o = I \oplus S_o$$
$$T_2 = T_1 \oplus S_i = (I \oplus S_o) \oplus S_i$$
$$R \;= (I\,/\,S_i) \oplus (S_o\,/\,S_i) = R_I \oplus R_S$$
$$\tilde{R} = M\,(R) = \tilde{R}_I \oplus \tilde{R}_S$$
$$T_3 = \tilde{R} \oplus S_i = (\tilde{R}_I \oplus \tilde{R}_S) \oplus S_i$$
$$T_4 = (\tilde{R}_I * S_i\,/\,\Sigma\tilde{R}) \oplus S_i = O \oplus S_i$$
$$T_5 = O \otimes (S_i \text{-} \Sigma O) = O \otimes S_o$$

Fig.6 *Categorical diagram of the basic kinon model*

An isolated kinon, in which respective input and output buffers are looped, is possible but the collective behavior of kinons organized in a network is far more interesting. Formally, a kinon network is a *balanced digraph*, i.e. a directed graph in which the in-degree and out-degree of every vertex $v_{i,}$ representing one kinon, are equal: $d^+(v_i) = d^-(v_i)$. A balanced digraph is said to be *regular* if all nodes have the same in-degree and out-degree. Zuse's net automaton and Pask's diffusion network, shown in Fig.1, exemplify regular and irregular kinon networks. Regular kinon networks, considered further, can be described by a node degree *d* and lattice width *w* (or a number of nodes *N*).

Motivation for the model extension

It was shown in the previous paper that the relational approach and innate tunability of the model, i.e. the ability to be controlled by a smooth variation of one or more real-valued parameters, dramatically increase the complexity of its behaviour in comparison to continuous cellular automata. Nevertheless, the only tunable block in the basic model is the modulator, while other blocks are firmly hardwired. Encoding and decoding blocks, performing trivial but very important transformations, also can be made tunable via the elaboration of their circuitry, and these enhancements may have crucial consequences for the model's dynamics.

According to Robert Rosen, encoding is closely related to the problem of measurement and can be stated by the following propositions (Rosen, 1978):

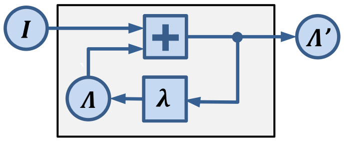
- *The only meaningful physical events which occur in the world are represented by the evaluation of observables on states.*
- *Every observable can be regarded as a mapping (or encoding) from states to real numbers*.

This view is in line with the approach to measurement of the American psychophysicist Stanley Stevens who defined measurement as *"the assignment of numerals to objects according to a rule"* (Stevens, 1946). Initially, he identified four levels of measurement defined by groups of scale invariant mathematical transformations: *nominal*, *ordinal*, *interval* and *ratio*, but later added another scale type, *log-interval* (Stevens, 1959). However, this list is not complete and ratio is not the ultimate level of measurement. The ratio scale has one fixed point ('zero') and the choice of the value of 'one' is essentially arbitrary. An *absolute* level of measurement can be obtained if the value of 'one' is also fixed. The most apparent example of the absolute scale is probability, where the axioms fix the meaning of 'zero' and 'one' simultaneously.

The kinon model was derived from the LBM model based on statistical mechanics; nevertheless, it is fully deterministic. It equates the value of 'one' to the total amount of storage and inflow in the kinon, but it is invariant only during the current cycle; therefore a scaling step of encoding is related to a ratio scale. On the other hand, a scaling block transforms absolute (raw) input values corresponding to a nominal scale. The usage of other scales or evaluation methods in encoding may contribute to the overall complexity of the model's behaviour.

For that purpose, additional structural elements corresponding to electronic analog filters can be added to the encoder, which will process raw input values (observables) before scaling. Such filter can be treated as *a meter* evaluating input values via a nonlinear map, e.g. logarithmic or other function with a domain and codomain in $\Re^+$. It will be a direct implementation of the Rosen's treatment of measurement as *'a mapping from states to real numbers',* or formally *f: S→ℜ.*

Another interesting option is the usage of a low-pass filter with memory known as *a leaky integrator*. A physical example of a leaky integrator is a bucket of water with a hole in the bottom. The rate of leakage is proportional to the depth of water depending on the difference between input and "leak" rates (hence the name). A very simple discrete time implementation of a leaky integrator is shown in Fig.7, where "*I*" stands for an input and "*Λ*" for a potential, which depends on the output in the previous time step and plays the role of water in a leaky bucket or short-term memory, which fades without reinforcement. This is akin to a moving average but does not require data buffering, which is very costly in computational and memory usage terms.



$$\Lambda' = I + \Lambda$$
$$\Lambda = \lambda\,\Lambda'$$
$$0 \le \lambda \le 1$$

Fig.7 *Discrete time leaky integrator (λ-filter)*

Leaky integrators find their use in the neural and cognitive modeling (Graben et al, 2008) and the modeling of systems with anticipation property (Makarenko et al, 2007). An anticipatory system is a system that contains an internal predictive model of itself and its environment, which allows it to change the current instant state in accordance with the model's predictions pertaining to a later instant (Rosen, 1985).

A leaky integrator is, perhaps, the simplest model capable of predicting the future state of a system and the easiest way to introduce a field approach in the kinon model. It can be implemented by linking additional variables to input buffers and storage, which will represent the channel potentials and play the role of local curvature or space. In this case, potentials will be influenced by input flow and, in their turn, influence output flow, representing matter. This is analogous to the famous quote by John Wheeler: *"Matter tells space how to curve and curved space tells matter where to move"*.

The usage of cut-off (threshold) filters for the elimination of unwanted marginal values or simulation of surface tension would be also beneficial for the overall model nonlinearity.

Extended model

A gathering block of the encoder was augmented by the embedding of λ-filters in all input and storage channels. They are leaky integrators with a common tunable real-valued control parameter λ in a unit range, representing a memory "leak" rate. A scaling block was enhanced by the incorporating of ψ-filters which transform input absolute values via a nonlinear function. The nonlinearity is vital here because modulation is scale invariant. This operation is similar to gamma-correction used in image processing for the adjustment of pixel intensity values according to human visual perception. The letter ψ is frequently used as a symbol of psychology and perception, so it was chosen for the name of the filter. The introduced filters are shown in Fig.8 and highlighted by a peach color.

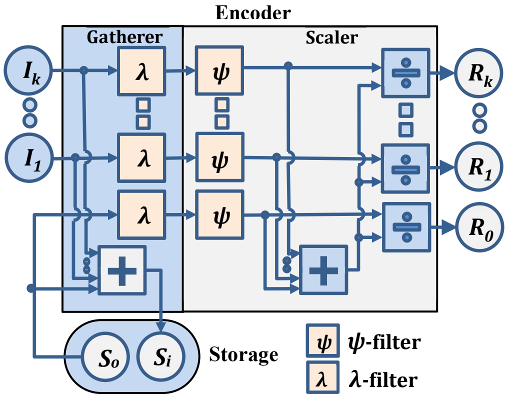


Fig.8 *Schematic of the tunable encoder*

Similarly, a decoding module, consisting of the rescaler and scatterer, was enhanced by adding two new kinds of analog filters shown in Fig.9. θ-filters truncate rescaled absolute output values below a threshold before scattering them into output buffers. They are tuned by a common real-valued control parameter θ in the range $[0, \Theta]$, where Θ is much less than the total quantity of the network. It can be treated as the simplistic simulation of fluid cohesion due to surface tension.

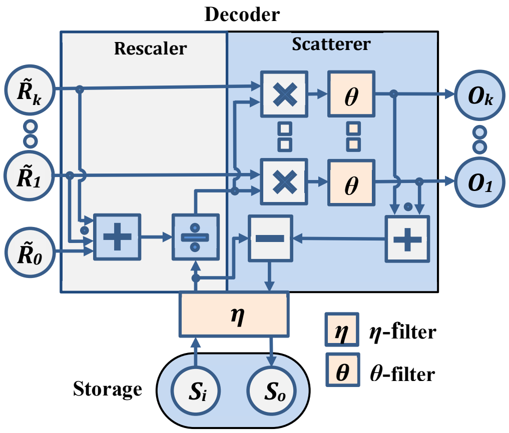


Fig.9 *Schematic of the tunable decoder*

Another novel filter, shown in Fig.10 in more detail, is analogous to λ-filter, i.e. a leaky integrator, but is a little different. Similar to a conventional leaky integrator, it integrates values obtained on different time steps, but in this case, both time steps take place during the same cycle. A tunable parameter η defines here not a storage 'leak' rate but a share of the input storage value not participating in distribution (decoding) and remaining in the storage. In medicine, a small passage which allows movement of fluid from one part of the body to another is called a shunt, so a shunting integrator is a more proper name for such a filter. It can be treated as the simplistic simulation of fluid viscosity, a quantitative measure of fluid resistance to flow drag, generally denoted by η. Its influence is similar to damping in mechanical systems, although it is not based on energy loss, so a damper might be another "telling" name for the η-filter.

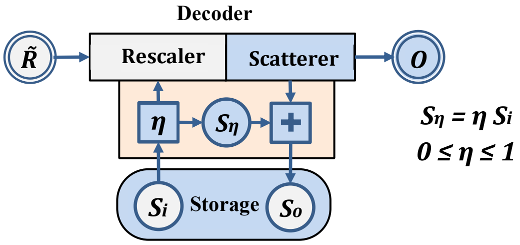


Fig.10 *Shunting integrator (η-filter)*

All introduced filters, denoted as f_λ, f_ψ, f_η and f_θ, with new variables corresponding to channel potentials (Λ, Λ'), their measurements (percepts) (Ψ, S_ψ) and fractions of input storage (S_η, S_δ) can be succinctly represented by the following categorical diagram of the extended kinon model [Fig.11], which represents *the full algorithm* of the kinon model. All alterations and additions against the categorical diagram of the basic kinon model [Fig.6] are shown in a red color.

$$\begin{tikzcd}
I \arrow[rrrrr, dashed, "\varepsilon"] \arrow[dr, "i_1"] & & & & & R \arrow[dddddd, "M"] \\
& T_1 \arrow[r, "i_1"] \arrow[ddr, "\Sigma"] & \Lambda' \arrow[r, "f_\psi"] \arrow[d, shift left, "f_\lambda"] & \Psi \arrow[r, "i_1"] \arrow[d, "\Sigma"] & T_2 \arrow[ur, "\psi"] & \\
& & \Lambda \arrow[u, shift left, "i_2"] & S_\psi \arrow[ur, "i_2"] & & \\
& S_o \arrow[uu, "i_2"] & S_i \arrow[d, "f_\eta"] \arrow[dr, "i_1"] & & & \\
& & S_\eta \arrow[ul, "i_2"] \arrow[r, "i_2"] & S_\delta \arrow[dl, "i_2"] \arrow[dr, "i_2"] & & \\
& T_5 \arrow[uu, "\pi_2"] \arrow[dl, "\pi_1"] & T_4 \arrow[l, "\theta"] & & T_3 \arrow[ll, "f_\theta \circ \eta"] & \\
O \arrow[uuuuuu, "P"] & & & & & \tilde{R} \arrow[lllll, dashed, "\delta"] \arrow[ul, "i_1"]
\end{tikzcd}$$

$$T_1 = P\,(O) \oplus S_o = I \oplus S_o$$
$$\Lambda' = T_1 \oplus \Lambda = (I \oplus S_o) + (\Lambda_i \oplus \Lambda_s) = \Lambda_i' \oplus \Lambda_s'$$
$$\Lambda = f_\lambda\,(\Lambda') = f_\lambda\,(\Lambda_i' \oplus \Lambda_s') = \Lambda_i \oplus \Lambda_s$$
$$\Psi = f_\psi\,(\Lambda') = f_\psi\,(\Lambda_i' \oplus \Lambda_s') = \Psi_i \oplus \Psi_s$$
$$T_2 = \Psi \oplus S_\psi = (\Psi_i \oplus \Psi_s) \oplus S_\psi$$
$$R = (\Psi_i / S_\psi) \oplus (\Psi_s / S_\psi) = R_I \oplus R_S$$
$$\tilde{R} = M\,(R) = \tilde{R}_i \oplus \tilde{R}_s$$
$$S_\eta = f_\eta\,(S_i)$$
$$S_\delta = S_i \oplus S_\eta = (S_i - S_\eta)$$
$$T_3 = \tilde{R} \oplus S_\delta = (\tilde{R}_I \oplus \tilde{R}_S) \oplus S_\delta$$
$$T_4 = f_\theta\,(\tilde{R}_I * S_\delta / \Sigma\tilde{R}) \oplus S_\delta = O \oplus S_\delta$$
$$T_5 = O \otimes (S_\delta - \Sigma O) = O \otimes S_\theta$$
$$S_O = \Sigma\,(S_\theta \oplus S_\eta)$$

Fig.11 *Categorical diagram of the extended kinon model*

Results

All results presented further are obtained using the simplest kinetic map: $y = Max[0, (1\text{-}\kappa x)]$. It was demonstrated in the previous paper that this map, despite its simplicity, exhibits highly complex behaviour with phase transitions. It has a single real-valued control parameter κ, which substantially simplifies the description of the parameter space and clarifies the influence of other parameters on the morphology of generated patterns.

Since this paper is aimed to show the applicability of the kinon model to morphogenesis, the evolution of the kinon network always starts from a 'singularity' state, corresponding to a zygote state in biological morphogenesis. It means that only one kinon has a non-zero storage equal to the total quantity (energy) of the network. This value is set to 20 000 for a 200 x 200 square grid, which is equivalent to 0.5 average value per a kinon visualized as a grey color in a greyscale image. It never changes during the evolution of the kinon network because quantity conservation is a staple feature of the model.

Apart from a kinetic map parameter κ in the basic model, the extended model has a set of additional parameters $\{\lambda, \psi, \eta, \theta\}$ controlling the introduced filters. The paremeters λ, η, θ are real-valued non-negative numbers equal to a zero by default, while ψ is a function with characteristic parameters. The latter parameter is equal to the identity map *id:* y=x if another is not specified.

It is not surprising that the most significant parameter is λ because it controls internal potentials relating to kinon 'memory' or 'anticipation' property. Just two parameters, κ and λ=*1*, are sufficient for the appearance of circular or square waves not observed in the basic model. The increment of the parameter κ results in the change of the shape and length of the wave and the speed of its final fission into four solitons [Fig.12]:

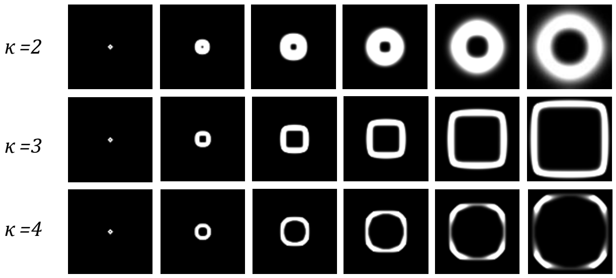


Fig.12 *Two-dimensional kinetic waves* (λ=*1*)

The adjustment of scattering θ-filters, which can be related to the change of surface tension, induces the appearance and development of manifold shapes. Using the same parameters κ and λ as above and the parameter θ=*2*, one can obtain dynamical patterns reminding the embryonic development of a 'tetrapod star fish' or a growth of a four-fold crystal [Fig.13]. The increments of the parameter κ result here in the dramatic changes of the final morphology of the creature.

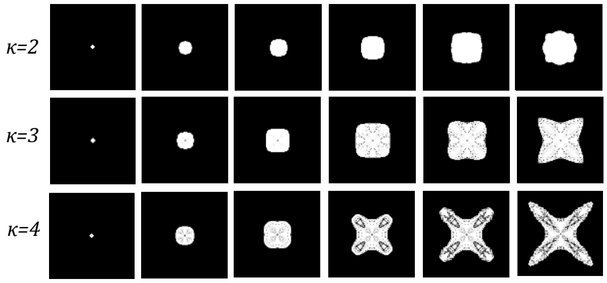


Fig.13 *Kinetic morphogenesis* (λ=*1* θ=*2*)

In all these examples, development starts from a single cell and goes through the stages which can be related to the oocyte, blastula, gastrulation and organogenesis stages in real biological morphogenesis. Due to the parameter θ, the developmental process finally stops at the state of a stable kinetic equilibrium corresponding to a mature full-fledged stage or stasis, which will be examined in the next paragraph in more detail.

For brevity, further demonstrations will show only the final stable state accompanied by a drawing of contour lines (isolines) after every 20 time steps (cycles). The results shown in Fig.14 demonstrate the influence of η-filter on the size and shape. According to a 'damping' nature of the η-filter, the final shape becomes more and more compact after the increment of this parameter.

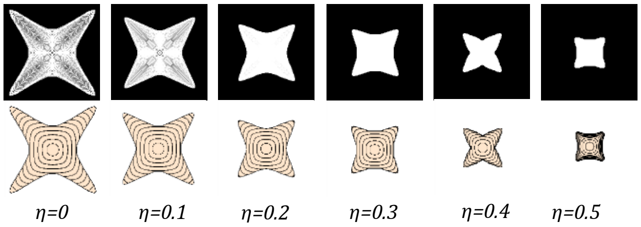


Fig.14 η-*patterns* (κ=*3* λ=*0.5* θ=*1*)

The influence of ψ-filter, which is closely related to the earlier discussed measurement problem and perception, cannot be so easily estimated. Similar to a kinetic map, it is a *functor* rather than a function, i.e. it maps functions but not values. The outcomes of the application of some ψ-filters under the same other parameters are shown in Fig.15.

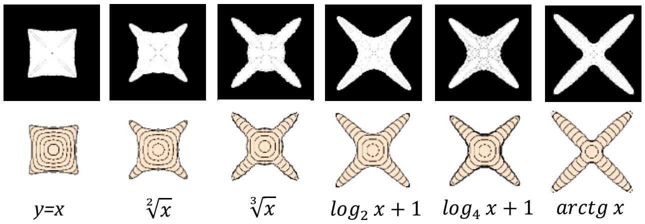


Fig.15 ψ-*patterns* (κ=*3* λ=*1* θ=*1.5* η=*0.1*)

It is evident that this filter affects not only the size but also the morphology of the resultant shape. Due to increasing nonlinearity of the function, the growing nucleus develops a branching structure that becomes more stretched and pointed.

The shown patterns were obtained using a square grid with four nearest orthogonal neighbors, i.e. a four-degree (*d4*) network. They have a distinct four-fold symmetry imposed by the underlying grid but preferential directions are aligned along the diagonals of the axes. This counter-intuitive phenomenon is related to the kinetic fission demonstrated in Fig.12, but is yet unexplained and needs further investigation.

The kinon model is topology invariant and allows arbitrary network structure including meshes and random networks. A *d4* network was chosen only for the ease of computation and visualization. A square grid with eight nearest neighbors (*d8*) requires extra computation per a cycle but is also readily visualized by a greyscale image.

Fig.16 demonstrates *ψ*-patterns in a *d8* network. The main difference from the previous experiment is the change of the preferred growth directions from diagonal to orthogonal. Besides, some new features of the body plan have appeared. Branches now may have not only pointed tips but also cleft and undulating ones.

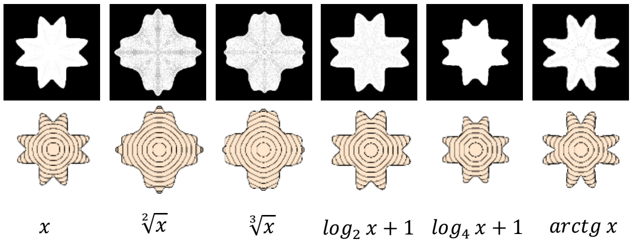


Fig.16 *d8 ψ-patterns (κ=8 λ=0.8 θ=0.3 η=0.4)*

The increased network connectivity makes kinon dynamics less predictable and parameter changes usually affect both the size and shape. Fig.17 demonstrates the susceptibility of the shape to minor variations of a single parameter, which indicates possible amenability of the extended kinon model to genetic and other evolutionary algorithms.

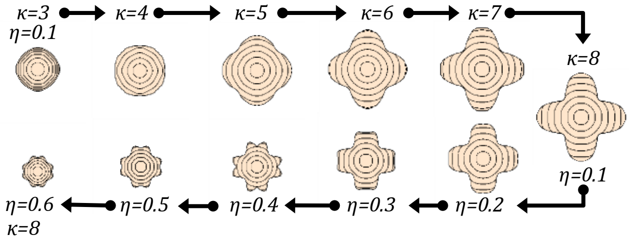


Fig.17 *Kinetic "metamorphosis" (d=8 λ=1 θ=0.6)*

The collections of some characteristic kinon morphogenetic patterns obtained using *d4* and *d8* grids are shown in Fig.18.

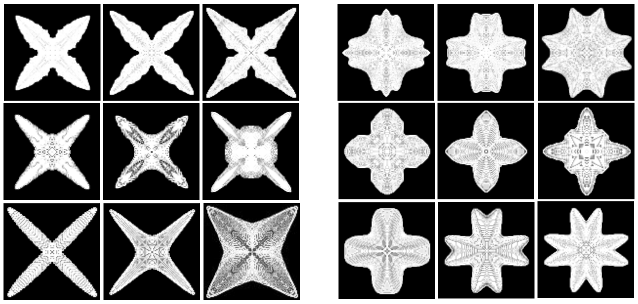


Fig.18 *Panopticon of d4 (left) and d8 (right) kinon creatures*

Speaking about a network structure, it is essential to define its boundary conditions. In practice, one cannot deal with an infinite lattice, so a common approach is to assume periodic (or cyclic) boundary conditions, i.e. to embed a lattice in a torus. However for morphogenetic studies, a better solution is the correct choice of the grid size and model parameters sufficient for a full-fledged state. Another option is the imposing of artificial boundaries (borders) on the network for the study of bounded growth. Due to topological invariance, the kinon model allows simple implementation of boundary conditions by the permanent elimination of respective links in the boundary kinons. The examples of bounded growth with different borders are shown in Fig.19.

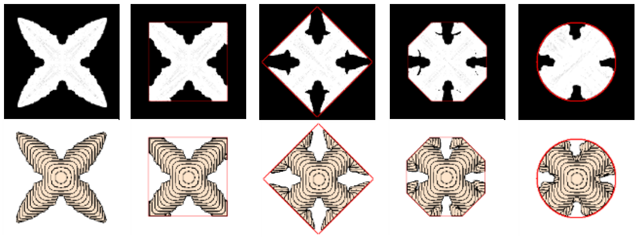


Fig.19 *Bounded growth (d=4 κ=6 λ=0.8 θ=0.4 η=0.5)*

Macrodynamic analysis

The total quantity of the kinetic network, which can be considered as energy or mass, is always the same because the model is conservative by definition. However, the total amount of all kinon output buffers (related to external or kinetic energy) and the total amount of kinon storage (related to internal or potential energy) interchangeably fluctuate. This feature can be used in the quantitative analysis of the kinon network macrodynamics.

The simplest macrodynamic index termed as an *exchange rate* K_e, is the ratio of the total value of all kinon output buffers to the total quantity of the network. It also can be interpreted as the ratio of the kinetic energy to the total energy of the network and is calculated as follows [Eq.1]:

$$K_e = \frac{\sum_{i=1}^{n}\sum_{j=1}^{k}\boldsymbol{O}ij}{\boldsymbol{\Omega}} \in [0,1], \qquad (1)$$

where: n - the number of kinons in the network;
k - the number of neighbors of the i^{th}-kinon;
$\boldsymbol{O}ij$ - the j^{th}-output buffer value of the i^{th}-kinon;
$\boldsymbol{\Omega}$ - the total quantity of the network.

Another macrodynamic index, also having a unit range and termed as *a turnover rate* K_t, is a half of the ratio of the absolute change of all kinon buffers during the current cycle to the total quantity of the network [Eq.2]:

$$K_t = \frac{\sum_{i=1}^{n}(|\Delta\boldsymbol{S}i| + \sum_{j=1}^{k}|\Delta\boldsymbol{E}ij|)}{2\boldsymbol{\Omega}} \in [0,1], \qquad (2)$$

where: $\Delta\boldsymbol{E}ij$ - the difference of respective exchange buffers ($\boldsymbol{I}ij$ - $\boldsymbol{O}ij$) of the i^{th}-kinon;
$\Delta\boldsymbol{S}i$ - the change of the i^{th} -kinon storage.

Even such simple macrodynamic indices can tell a lot about the current state and dynamics of the whole network [Fig.20]:

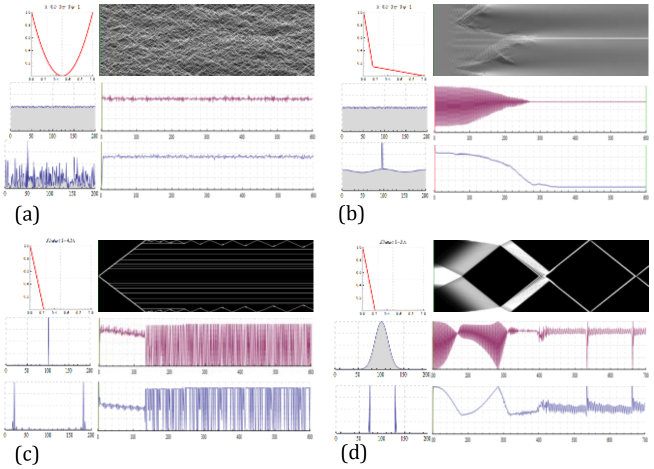


Fig.20 *Macrodynamics of one-dimensional kinon networks*

The first rows in these examples show the kinetic maps and visual images of the dynamics of one-dimensional kinon networks, which initial and ending states are represented by histograms in the left below. The plots below images represent the dynamics of exchange rates *Ke* and turnover rates *Kt* depicted in red and blue colors respectively.

Fig.20a shows the dynamics in a chaotic state, while fig.20b demonstrates the transition from a nearly equilibrium state to a stable non-uniform state. Both examples start from the same initial state but show totally different behaviour. Despite the final visual stasis in the latter figure, both indices attain a constant non-zero value. It means that the kinon network reached a coherent dynamic state, which can be related to Nash equilibrium in game theory. Fig.20c displays an almost still picture after the collision of solitons, sharply contrasting to the behaviour of both indices, oscillating in a nearly full range. Fig.20d illustrates the behaviour of these indices during the fission of the bell-shaped expanding wave in two solitons and their subsequent collisions.

Macrodynamics of some characteristic morphogenetic patterns is represented in Fig.21. The green line marks the beginning of stasis and the orange line marks the time step when dendrite tips hit the border.

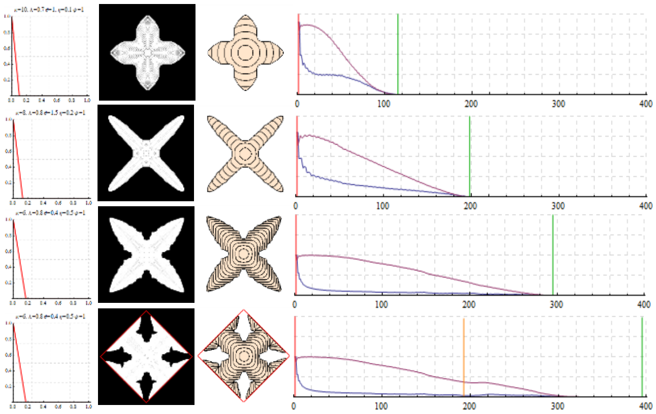


Fig.21 *Kinetic macrodynamics of morphogenesis*

These indices supplement visual representation of the kinon network dynamics and are indispensable in cases when visualization is intractable or the automation of parameter space exploration is needed, e.g. for the search of viable stable shapes. All demonstrated morphogenetic examples come to a matured full-fledged stable state (stasis) in which both macrodynamic indices reach a zero value.

Discussion

The shown results demonstrate that the extended model, employing only trivial math, is capable of producing complex patterns, some of which resemble crystal dendrites. Dendritic crystal growth is very common and may be illustrated by snowflake formation and frost patterns on a window. In metallurgy, a dendrite is a characteristic tree-like structure of crystals growing during molten metal solidification. This dendritic growth has large consequences in regard to material properties. That is why much research has been devoted to the simulation of crystal growth, and one of the most employed approaches is a phase-field model. Despite the enormous progress in computational terms, a phase-field model, based on the Ginzburg-Landau equation, still requires quite hard computation. The kinon model was not tailored to simulate dendritic growth in melts; nevertheless, it captures this phenomenon rather closely and generates very similar shapes [Barrett et al, 2014].

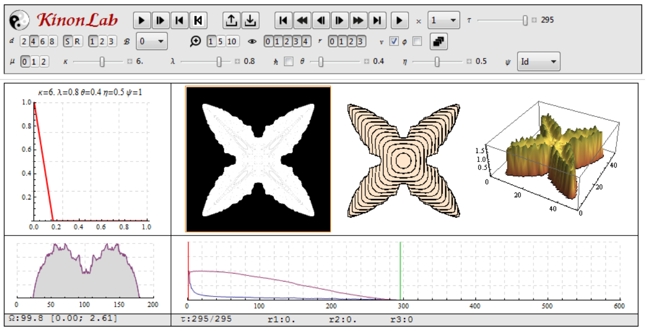
Pattern generation and morphogenesis are not restricted to physical and biological systems and may be realized in many other contexts. In Robert Rosen's words: *"Morphogenesis, in the widest sense, is the generation of pattern and form in a population of interacting elements".* He showed that morphogenetic techniques can be also applied to human settlement patterns and the emergence of cultural differentiation in a human population (Rosen, 1979). He was, possibly, the first to assert that active transport of materials against their diffusion gradients is ubiquitous in biology and to show that the coupling of reactions and diffusion can result in surprising effects, such as the accumulation of population against density gradients. The coupling of diffusion with reaction can play the role of specific "pumps" moving populations uphill their gradients.

Moreover, Rosen envisaged the possibility of asymptotically stable non-uniform states with just one morphogen, rather than two as in Turing's original paper, which was recently rediscovered by the author using a different approach. Turing implicitly assumed that each cell was homogeneous and isotropic with respect to diffusion in all directions. Rosen showed that it is perfectly possible to construct a simpler system by removing the assumption of diffusional isotropy (Rosen, 1981).

The kinon model proved to be able to produce complex shapes using very simple local interactions by the exchange of real-valued numbers, which can be regarded as the quantities of the elusive long-sought chemical signal, which Turing called a morphogen. A large collective of kinons embodied in a robot swarm would gradually aggregate into dendritic or more complex structures by the exchange of such morphogen with the nearest neighbors.

Contrary to another kinetic critical phenomenon called diffusion-limited aggregation (DLA) (Witten & Sander, 1981), this model generates totally deterministic rather than random shapes. The kinon model does not need a global clock, but does require local synchronization. Unlike static regular kinon networks, considered in this paper, a swarm network topology is irregular in a general case and node neighborhoods are variable. In order to cope with these circumstances, the necessary mechanisms for local interaction and synchronization must be added to the propagator.

All figures demonstrating kinon network states and dynamics [Fig.12-21] were obtained using a framework called *KinonLab*, developed by the author. It is implemented in Wolfram *Mathematica*® and currently supports one-dimensional (*d2*) and two-dimensional regular networks with von Neumann (*d4*) and Moore (*d8*) neighborhood [Fig.22].



× 1 τ 295

κ 6. λ 0.8 θ 0.4 η 0.5 ψ Id

Ω:99.8 [0.00; 2.61]

τ:295/295 r1:0. r2:0. r3:0

Fig.22 *KinonLab screenshot*

The main directions of the ongoing improvement include the support of two-dimensional hexagonal (*d6*) and three-dimensional regular grids, and also the development of tools for micro-dynamic and comparative analysis. The future developments of the framework will be aimed at the study of *multi-component* and *arbitrary topology* kinon networks.

Conclusion

The presented here extended kinon model demonstrates its high potential for structural elaboration and the need for deeper exploration and validation. The given formal definition clarifies some ambiguities of the previous description and provides a common language and terminology to facilitate its further development. The schematic and categorical diagrams of the kinetic automaton not only illustrate its operational structure ('modus operandi') but also show the directions of the possible implementation of the model with analog circuits. The proposed macrodynamic indices provide the expressive measures for the quantitative analysis of kinon network dynamics and parameter space exploration.

The main aim of this paper is to demonstrate the applicability of the model to the problem of morphogenesis. The shown results confirm that the extended model is capable of producing spatio-temporal patterns pertaining to morphogenesis in real physical and biological systems. In respect to Artificial Life research, the most promising directions of the kinon model application are rapidly emerging fields of morphogenetic engineering (Doursat et al, 2013) and swarm robotics (Brambilla et al, 2013).

Acknowledgments

I would like to thank Dr. Aloisius H. Louie for his expert advice and warm encouragement during the elaboration of the categorical kinon diagrams.

References

Barrett, J.W., Garcke, H., Nuernberg, R., (2014) Stable Phase Field Approximations of Anisotropic Solidification, *IMA Journal of Numerical Analysis*, 34: 1289-1327

Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M. (2013) Swarm Robotics: a Review from the Swarm Engineering Perspective. *Swarm Intelligence* 7 (1), 1-41

Chopard, B., Luthi, P., Masselot, A. (2002) Cellular Automata and Lattice Boltzmann Techniques. An Approach to Model and Simulate Complex Systems. *Advances in Complex Systems,* 5:2, 103-246

Doursat, R., Sayama, H., Michel, O., (2013) A Review of Morphogenetic Engineering. *Natural Computing*, 12(4), 517-535

Graben, P.B., Liebscher, T., Kurths, J. (2008) Neural and Cognitive Modeling with Networks of Leaky Integrator Units. In *Lectures in Supercomputational Neurosciences. Understanding Complex Systems*. Springer-Verlag Berlin, 195-223

Kauffman, S.A. (2000) *Investigations*. Oxford University Press.

Louie, A.H. (1983) Categorical System Theory. *Bulletin of Mathematical Biology* 45(6), 1047-1072.

Makarenko, A., Goldengorin, B., Krushinski, D. (2008) Game 'Life' with Anticipation Property. In: *Proceedings of ACRI 2008*. LNCS, vol. 5191, Springer, Heidelberg, 77–82

Pask, G. (1961) A Proposed Evolutionary Model. Foerster von, H., Zopf, G. Eds. *Principles of Self-Organisation*. Pergamon Press, 229-254

Rosen, R. (1978) *Fundamentals of Measurement and Representation of Natural Systems*. New York: Elsevier North-Holland.

Rosen, R. (1979) Morphogenesis in Biological and Social Systems. In: Renfrew, A.C., Cooke, K.L. (eds.) *Transformations: Mathematical Approaches to Culture Change,* Academic Press, London, 91-112

Rosen, R. (1981) Pattern Formation in Networks. In: *Progress in Theoretical Biology*, Vol. 6 Academic Press. 161-209.

Rosen R. (1985) *Anticipatory Systems: Philosophical, Mathematical & Methodological Foundations*. New York: Pergamon Press.

Rosen, R. (1991) *Life Itself: A Comprehensive Inquiry into the Nature, Origin, and Fabrication of Life*. NY: Columbia University Press

Shalygo, Y. (2014) The Kinetic Basis of Self-organized Pattern Formation. In *Proceedings of ALIFE 14*, MIT Press, Cambridge, MA, 665-672 http://dx.doi.org/10.7551/978-0-262-32621-6-ch106

Stevens S.S. (1946) On the Theory of Scales of Measurement. *Science*, 103, 677-680

Stevens S.S. (1959) Measurement, Psychophysics and Utility. In Churchman C.W., Ratoosh P. (Eds). *Measurement: Definitions and Theories*, New York: Wiley, p.18-63

Turing, A.M. (1952) The Chemical Basis of Morphogenesis. *Phil. Trans. of the Royal Society of London*, B 237:37-72

Witten Jr., T.A., Sander, L.M. (1981) Diffusion-Limited Aggregation, a Kinetic Critical Phenomenon, *Phys. Rev. Letters*. 47, 1400

Zuse, K. (1969) *Rechnender Raum*. Braunschweig: Friedrich Vieweg & Sohn [English translation: Calculating Space. MIT Technical Translation AZT-70-164-GEMIT, MIT Cambridge MA Feb. 1970]

Environmental Factors and the Emergence of Cultural – Technical Innovations

Peter Andras[1]

[1]School of Computing and Mathematics, Keele University, UK
p.andras@keele.ac.uk

Abstract

Environmental factors that determine ecological niches, for example natural boundaries formed by mountains, rivers, deserts, contribute to the speciation among animals. Similar factors have been proposed to be important for the emergence of cultural and technical innovations in human populations in the pre-state stages of societies. Here we describe a social simulation aimed to investigate this issue. The simulation uses two environmental features, mountain ridges and the fertility of the land. The results show that indeed these environmental factors matter for the emergence of successful innovative populations. The defenses provided by mountain ridges facilitate the emergence of many populations with moderately successful innovations. The fertile lands are where the populations with the most successful innovations emerge, however in most cases these populations can trace their origins to innovative populations emerging under the defense of mountain ridges. This simulation study provides experimental support for the relatively speculative theories about the importance of environmental factors for the emergence of cultural and technical innovations.

Introduction

The emergence of cultural – technical innovations in human populations determined the dynamics of locally and globally dominant cultures across history (Bouckaert et al, 2012; Chiaroni et al, 2008; Diamond and Bellwood, 2003; Mellars, 2006). The factors that promote the emergence of such innovations are not well established and there are several theories that aim to explain these events (Diamond, 1997; Diamond and Bellwood, 2003; Fukuyama, 2014). In general such innovation give a competitive advantage to the populations that adopt them in the context of their physical and social environment, making these populations grow faster than others or have better chance of winning in battles – consider for example the emergence of agriculture, the domestication of cows and horses, the flexible use of fast light cavalry by migratory populations in Europe and Asia in the middle ages, or the introduction of cannons in the late middle ages in Europe, Asia and North Africa (Diamond and Bellwood, 2003; Fukuyama, 2014; Mellars, 2006).

Considering early human populations some areas of the Earth stand out as relatively frequent sources of major waves of culturally – technically dominant populations and others stand out as the most culturally diverse areas. For example, the area between the Altay Mountains and the Fergana Valley in Central Asia has been the source of several waves of radiation of culturally – technologically dominant populations that went on to control, rule and dominate large parts of the world – e.g. Mongols, Turkic populations that established empires in Western and South Asia and regularly threatened or even conquered China (Fukuyama, 2014). Examples of high linguistic and cultural diversity include the highlands of Papua New Guinea (Reesink et al, 2009) and the Caucasus (Bulayeva et al, 2003).

In the case of animals and plants speciation is driven by the emergence of functional innovations that give a competitive fitness advantage to individuals having these innovative features in the context of some ecological niche (Barraclough et al, 1998; Warren et al, 2008). Examples include shrinking of size in island populations of animals, ability to access and digest new sources of food, or the emergence of the ability to glide and fly. In these cases the physical boundaries (e.g. large mountains or rivers, deserts) and features of the environment (e.g. humidity, aridity, temperature, light conditions) often play an important role in the definition of the environmental niche where the functional innovations may convey fitness advantage (Barraclough et al, 1998).

It has been proposed that environmental factors, such as mountain and rivers barriers, land fertility, presence of disease vectors (e.g. mosquitoes, flies), matter for the likelihood of emergence of cultural – technical innovations in human populations (Fukuyama, 2014). For example, many relatively isolated and hard-to-access valleys in highland areas may facilitate the maintenance of many separately developing populations providing them sufficient protection from neighbors to develop and perfect potentially advantageous cultural and technical innovations. Or, proximity to fertile lowland areas may facilitate the expansion and the development of further advantageous innovations in growing populations emerging from protected valleys, which already have some cultural –technical advantage over other neighboring populations.

However, these theories and proposals rely to a good extent on speculations constrained by the available historical and archaeological evidence. Naturally, there is relatively little objective evidence to confirm these theories. One possible approach for gathering more supporting evidence is to build simplified simulations of evolution and spreading of human populations including the simulation of emergence of cultural – technical innovations and test assumptions about the roles of environmental factors through analysis of the outcome of these simulations.

Here we describe a simulation of evolution and spreading of human populations, including the simulation of natural boundaries as mountain ridges and the variability of the land fertility as environmental factors with expected impact on the emergence of cultural – technical innovations. The simulated human populations are enabled to generate cultural – technical innovations that change their fitness for growth. We analyzed the impact of the considered two environmental factors on the emergence of these innovations by considering populations that are successful in maintaining themselves over considerable simulated time and in achieving sufficiently large relative size among the simulated populations.

Our results show that the proximity of mountain ridges that protect from conquest by neighbors and the low land fertility of the area correlate with innovations that confer large positive changes in the competitive fitness of the simulated human populations. The results also show that the most dominant populations originate from fertile lowlands, however in all cases the roots of these populations go back to places close to mountain ridges. Overall, the simulation results confirm the importance of the considered environmental factors for the emergence of cultural – technical innovations.

The rest of the paper is organized as follows. First we review the related research. Then we describe in detail the simulation of evolution of human populations constrained by environmental factors. Then we present the simulation results and their analysis. The paper is closed by a discussion and conclusions section.

Related Works

The emergence and evolution of animal and plant species and their spreading across areas of the Earth has been researched for long time. In general, ecological niches provide the setting for the emergence of new species and the species spreads territorially and to other ecological niches by outcompeting other species resident in these spaces (Barraclough et al, 1998; Warren et al, 2008). The role of environmental barriers, such as mountain ridges, deserts, sea, large rivers and lakes and vast and dense forests, has been studied in the context of ecological niche formation (Barraclough et al, 1998; Warren et al, 2008).

The evolution of human populations and the emergence of cultural or technical innovations that give an advantage to a population in comparison with neighboring populations have been considered as an intriguing question by many researchers (Diamond, 1997; Fukuyama, 2014). There is increasing volume of data, e.g. archaeological data, population genetics data, linguistic data, that can be used to support theoretical proposals in this area of research (Bouckaert et al, 2012; Chiaroni et al, 2008; Der Sarkissian et al, 2013; Diamond and Bellwood, 2003; Reesink et al, 2009). However, the limited nature of all these data means that often it is difficult to decide about the validity of these theoretical proposals.

In general it is assumed by many researchers that environmental factors such as well separated highland valleys, or alluvial valleys of large rivers, or lack or abundance of fertile land, contribute to the chances of human populations inhabiting such areas to survive, become dominant in a larger area, and to develop successful cultural or technological innovations (Diamond and Bellwood, 2003; Fukuyama, 2014; Mellars, 2006). For example, protected highland valleys may allow the co-existence of many culturally / technologically different human populations, which may try out many potentially successful innovations. Some of these innovations may confer sufficient competitive advantage to the inventor population to conquer other populations over extended areas that offer less natural protection if such areas are available not too far from the original location of the inventor population. Some argue about the importance of alluvial valleys or fertile land for the expansion of the population, which is required as a pre-requisite for the later conquest of other populations and areas (Fukuyama, 2014).

The modeling of spreading of populations has a well established mathematical theory rooted in the study of reaction – diffusion equations (Garcia-Ramos and Rodrigues, 2001; Petrovskii et al, 2002). The general equation of this theory describes the spatio-temporal diffusion of the population combined with a reaction term that represents the impact of external factors, for example the local growth of the population as a function of available resources and efficiency of use of these resources. The equation is as follows:

$$\partial u(x,t)/ \partial t = \nabla(D(u,x) \cdot \nabla u(x,t))+F(u,x,t) \qquad (1)$$

where $u(x,t)$ is the size of the population at spatial position x and at time t. The term $D(u,x)$ describes the diffusion of the population at location x, and $F(u,x,t)$ is the reaction term that describes the change in the size of the population at spatial location x and at time t due to the external factors. In general this equation cannot be solved, but solutions of particular cases can be found. Another approach is to solve step-by-step a discretized version of the equation.

It has been shown that population dispersion described through equation (1) leads to wavefront propagation phenomena and Turing patterns of waves (Cheng et al., 2014; Jeltsch et al, 1997; Lambin et al, 1998). The wavelength of these patterns depends on the diffusivity of the population, the quicker is the diffusion the shorter is the wavelength of the patterns, and in general the wavelength is proportional to the square-root of the diffusivity value (Ouyang et al, 1995).

Simulation based modeling of population spreading follows from the discretized version of equation (1). Computer simulations have been developed to model particular cases of population spreading (e.g. spreading of certain invasive species) (Cheng et al, 2014; Garcia-Ramos and Rodrigues, 2001). The simulations often assume homogeneous diffusivity, i.e. the same diffusivity everywhere. The assumptions about the reaction term are based either on some reproduction and death dynamics (i.e. a combination of adding and subtracting a number of individuals on the basis of an equation describing this population size dynamics) or on interactions with one or more other species. The simulations of population spreading produce the expected Turing patterns of traveling waves (Cheng et al., 2014; Petrovskii et al, 2002). The Turing patterns are generated in a uniform manner, as the reaction terms usually apply uniformly across the whole simulated space. The patterns observed in the simulations are similar to the actual observation of spreading of species in cases where sufficient data is available (Jeltsch et al, 1997; Lambin et al, 1998).

The spreading of innovations across populations has been studied for decades (Mahajan and Muller, 1979). These works focus on how innovations spread through adoption by populations and the results are similar to the spreading of infectious diseases (Delre et al, 2010; Macy and Willer, 2002). In this paper we do not consider the adoption of cultural and technical innovations by imitation across populations.

Simulation of the Evolution of Human Populations

The simulated world is a 2-dimensional grid of 100×60 spatial locations. Each location has a height and a harshness value. The height gives the altitude of the location and the harshness is the opposite of the land fertility. The height value is an integer between 0 and $g_{max} = 100$. The harshness value is a real value between 0 and 1, larger value meaning harsher, less fertile, spatial location. The height values are set such that the locations arranged along lines have the same height value – these locations together form a simulated mountain ridge. Each simulated world has $m = 20$ mountain ridges of variable length. Each simulated world also has $n = 10$ low fertility or harsh areas as well. The low fertility areas are circular areas set such that their centre is located on one of the mountain ridges. At the centre of these areas the harshness value is 1 and it is decreasing proportionally with the square of distance from the centre. The radius of low fertility areas is set randomly between $d_0 = 50$ and $d_{max} = 100$. If a location belongs to multiple low fertility areas the harshness value of the location is the sum of the harshness values implied by each low fertility area to which the location belongs, capped at the maximum value of 1. Figure 1 shows an example of the simulated world with mountain ridges and low fertility areas.

Each simulation starts with a random number of simulated human populations, such that a proportion around $\chi = 1.5\%$ of all spatial locations have exactly one distinct human population on them, while all others are empty at the start, i.e. the number of simulated populations in our simulations is around 90 at the beginning.

Each population is characterized by a set of abstract cultural and technical features represented by an ability string A of 0-s and 1-s, where the 1-s indicate the presence of a such feature. Here we chose the length of ability strings to be $L = 100$. The ability string determines the resource utilization efficiency of the population. If $A=(a_1,...,a_L)$ is the ability string of a population then the resource utilization efficiency of this population is

$$r=\Sigma_{k=1,L1}\Sigma_{j=1,L2}\, a_{L1\cdot(k-1)+j}\cdot 2^j \qquad (2)$$

where $L1 \cdot L2 = L$ and in our case $L1 = L2 = 10$. This definition of resource utilization efficiency allows multiple cultural and technical features to contribute equally to the efficiency of the population, which is the case of real world human populations.

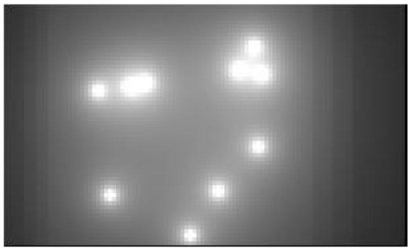


Figure 1. Distribution of land fertility in a simulated world. Brighter locations indicate harsh, low fertility land, darker locations indicate less harsh, more fertile land.

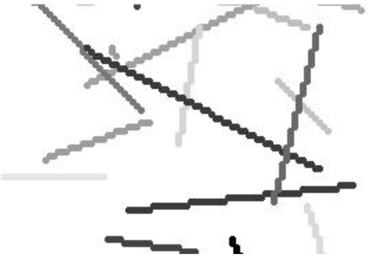


Figure 2. Mountain ridges in a simulated world. The darkness indicates the height of the ridge, darker locations are higher.

Each population may be spread over a number of space locations and each space location may house representatives from at most N_{max} (in our case this is 8) populations. The resource availability at each location is limited and is characterized by the harshness value of the location. Each population at each location have a basic death rate θ which is set in our case to 0.005. Each location has a maximum population size limit for each population residing in that spatial location and this depends on the resource utilization efficiency of the population and the harshness of the land at this location. The maximum size of a population with resource utilization efficiency r at a location (x,y) with harshness $h(x,y)$ is assumed to increase logarithmically with r and to be inversely proportional with the harshness and it is set to be

$$p_{max}=(10+ln(1+r))\ /\ h(x,y) \qquad (3)$$

The populations at a given space location compete for the use of resources. The growth rate of the population at a space location is determined by the resource utilization efficiency of the population, the land fertility of the location, and the competition with other populations in terms of resource utilization efficiency. The growth rates of the populations are assumed to grow logarithmically with the resource utilization efficiency, to be inversely proportional with the harshness of the area and to be proportional with the level of competitiveness of the population. The death rate is considered as a subtracted element in the growth rate calculation. Assuming that there are N different populations at a given spatial location (x,y), and r_i, $i=1,...,N$ are the resource utilization efficiency values for these populations the growth rates of populations are calculated as

$$\rho_i(x,y)=\gamma\cdot(1+0.1\cdot ln(1+r_i))\ \cdot\eta_i\ /\ h(x,y) - \theta \qquad (4)$$

where $h(x,y)$ is the harshness value at location (x,y), η_i are the competitiveness factors, θ is the basic death rate. The competitiveness factor for each population depends on the ratio between the resource utilization efficiency of the considered population and the sum of the resource utilization efficiencies of all present populations and follows a sigmoidal curve as a function of this ratio, i.e. saturating both for very and high and very low values of the this ratio. For this simulation η_i are calculated as

$$\eta_i=1/(1+exp(\alpha\cdot(0.5-r_i\ /\ (\Sigma_{k=1,N}\, r_k))) \qquad (5)$$

γ and α are parameters characterizing the default population growth and the level of competition – our choices were $\gamma=1.4$ and $\alpha=10$. Higher values of γ imply quicker default population growth, higher values of α imply more competition between the populations.

If the size of a population at a given location grows beyond the allowed maximum size for the population at this location, the population size gets capped at the maximum allowed size. If the population size goes below zero the population disappears from this location. If the number of population at a location goes above the maximum limit for the number of different populations at a given location, the population with the smallest size gets removed.

The simulated human populations generate cultural / technical innovations randomly with $\varepsilon = 0.008\%$ chance in each time turn of the simulation – we assume that there is no cost associated with the development of these innovations. These innovations are implemented as a random flipping of cultural / technical feature indicator in the ability string A from 0 to 1 or reverse. This naturally changes the resource utilization efficiency of the newly generated descendant population in comparison with the resource utilization efficiency of the parent population. A proportion of the parent population converts to the descendant population and this proportion is determined by the two resource utilization efficiency values in a similar manner that we used above for the calculation of the competitiveness of populations. If r_p and r_d are these values for the parent and descendent populations first we determine

$$\eta_{p/d}=1/(1+exp(\alpha\cdot(0.5-r_{p/d}\ /\ (r_p+\ r_d))) \qquad (6)$$

then the ratio of the population converted to the new population is

$$w=\eta_d\ /(\eta_p\ +\eta_d) \qquad (7)$$

The populations spread across the simulated world by moving from one space location to neighboring space locations, i.e. through a diffusion process. The spreading from one location to a neighboring location depends on the height value of the location from which the spreading may occur. Let us consider $Q_i(x,y,t)$ the number of individuals belonging to population i at location (x,y) at time turn t, the equation for the update of $Q_i(x,y,t)$ is the following

$$\begin{gathered} Q_i(x,y,t+1) = \\ \Sigma_{(\tau,\upsilon)\in T}(\varphi_{x+\tau,y+\upsilon,-\tau,-\upsilon,t,i}\cdot Q_i(x+\tau,y+\upsilon,t) - \varphi_{x,y,\tau,\upsilon,t,i}\cdot Q_i(x,y,t)) \\ +\rho_i(x,y)\cdot Q_i(x,y,t) \end{gathered} \qquad (8)$$

where $T=\{(-1,0),(1,0),(0,-1),(0,1)\}$, $\varphi_{x+\tau,y+\upsilon,-\tau,-\upsilon,t,i}$ and $\varphi_{x,y,\tau,\upsilon,t,i}$ are stochastic diffusivity parameters, and $\rho_i(x,y)$ is the growth rate calculated according to equation (4). The stochastic diffusivity parameters are set as follows

$$\begin{gathered} \varphi_{x+\tau,y+\upsilon,-\tau,-\upsilon,t,i} = \psi \quad if \\ Q_i(x,y,t) > max\{g(x+\tau,y+\upsilon),g(x,y)\} \\ otherwise \\ \varphi_{x+\tau,y+\upsilon,-\tau,-\upsilon,t,i} = 0 \end{gathered} \qquad (9)$$

where $g(x+\tau,y+\upsilon)$ is the height value at location $x+\tau,y+\upsilon$, with $(\tau,\upsilon)\in T\ \cup\ \{(0,0)\}$ and ψ is a random value from the interval $[0,\omega]$ with $\omega<1$ (in the simulations we used $\omega=0.4$). This setting means that the populations can spread into the neighboring spatial locations either to the left or right or to up or down relative to the current location of the population and in any of these directions spreading happens only if the population is sufficiently large in comparison with the height of the current and neighboring locations. Note that the ω parameter influences the speed of diffusion (or diffusivity) of the populations, larger ω implying quicker spreading of the populations.

As described above the simulation starts with a number of distinct populations (around 90), which spread around in the simulated world. Populations compete with each other for the use of resources, the available amount of which is determined by the level of harshness of the spatial location (the opposite of land fertility). The spreading of populations is constrained by the presence of simulated mountain ridges. Spreading across such mountain ridges is difficult and the difficulty rises with the height of the simulated mountains. The populations randomly spawn new populations which are characterized by a cultural – technical innovation that changes their resource utilization efficiency compared to the parent population. In general these innovations may increase or may decrease the resource utilization efficiency. The simulated world evolves through a large number of time turns (we used $30{,}000$ time turns for each simulation).

The simulations are analyzed in order to determine the dominant populations. The dominance of a population is given by the size of the population across all spatial locations relative to the size of all populations across all spatial locations of the simulated world, i.e.

$$\pi_i(t) = (\Sigma_{(x,y)}\, Q_i(x,y,t))\ /\ (\Sigma_i\Sigma_{(x,y)}\, Q_i(x,y,t)) \qquad (10)$$

All populations for which $\pi_i(t) > \pi_0 = 0.005$ are considered sufficiently dominant populations and considered for further analysis, i.e. these populations represent at least half percent of the total inhabitants of the simulated world.

We aim to test three hypotheses based on the observation of spreading of animal and human populations in the context of environmental constraints.

Hypothesis 1: Being close to mountain ridges offers relative protection from invasion by other populations and allows the relatively frequent development of innovations with positive effect on the resource utilization efficiency in populations residing in such spatial locations. It is expected that the closeness to simulated mountain ridges in general allows the relatively long survival of innovating populations with both useful and harmful innovations. Out of these populations the ones with useful innovations are expected to have sufficient time to grow and then expand elsewhere due to the protected nature of their origin location.

Hypothesis 2: Existence at locations with low land fertility (harsh locations) facilitates the development of significant positive innovations. It is expected that if innovations occur in populations living in harsh conditions these innovations must be considerably positively useful in order for the innovating population to survive. Innovating populations that have harmful innovations from the perspective of their resource utilization efficiency are expected to die out quickly in the harsh environment. In such environment it is also expected that populations with useful innovations quickly outcompete populations which have lower resource utilization efficiency.

Hypothesis 3: The most dominant populations originate from locations with high land fertility (low harshness). It is expected that populations expand most quickly in high land fertility environments and developing useful innovations in such environments equips already large populations with the useful innovations which spread even more extensively than their parent population. On the other side it is also expected that populations that gain dominance on in high fertility environments must have had an earlier time when they benefited from the development of successive innovations relatively unperturbed by invaders, in line with Hypothesis 1.

To test these hypotheses we track all populations and select for further analysis those that are sufficiently dominant (see equation (10)). For the selected populations we consider their location of origin and the location of origin of their ancestors. For these locations we consider the harshness of the location and the distance of the location from the closest simulated mountain ridge. Considering the populations along their existence we measure the length of existence of the population (population persistence time), the peak share of the population out of all inhabitants of the simulated world, and the value of the innovation that sets the population apart from its immediate ancestor populations (i.e. the difference between the resource utilization efficiency of the two populations). Following the simulation we analyze the relationships between the ridge distance and harshness of the locations of origin of the considered populations and the measured features of the populations. The analysis is repeated for multiple (10 – 20) simulations to gather large volume of data. We also varied the values of γ and α which characterize the default population growth and the level of competition among populations residing at the same spatial location in order to assess the impact of these parameters on the evolution of the simulated populations.

Results and Discussion

Each simulation was run for 30,000 time turns. The simulation parameters were set as indicated in the description of the simulations in the previous section. Each population was identified by an identification number. For each time turn we recorded the details of all extant populations, including their location of origin, the identification number of their immediate ancestor population, the ability string of the population, the total size of the population across all spatial locations and the share of the population out of the total number of inhabitants in the simulated world at the given time step. The primary analysis of the simulation data was performed as described in the previous section in order to gather the information relevant for the testing of the hypotheses.

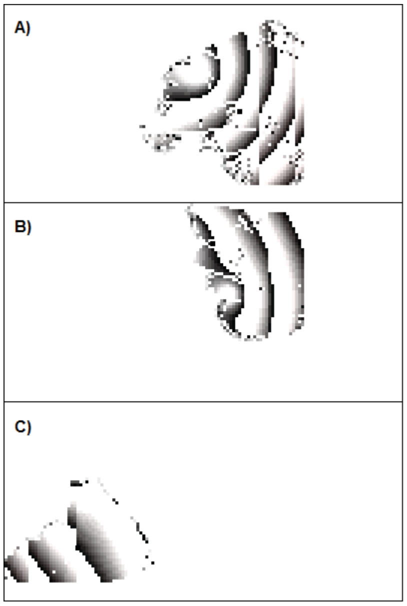


Figure 3. Turing patterns of population spreading in the simulated worlds for different values of the diffusion parameter ω: A) $\omega=0.9$ – fast spreading; B) $\omega=0.4$ – medium speed spreading; C) $\omega=0.03$ – slow spreading. White indicates no presence of the considered population, the level of darkness indicates the extent of population presence, darker meaning the presence of higher number of individuals belonging to the considered population.

The simulations show that patterns of spreading of populations follow Turing patterning. Varying the diffusion speed parameter ω we can vary the wavelength of the populations follow Turing patterning. Varying the diffusion speed parameter ω we can vary the wavelength of the population spreading patterns, which according to the theory should be proportional with the square root of the local diffusivity value (Ouyang et al, 1995). Note that given the presence of simulated mountain ridges and low land fertility areas the diffusivity of the populations varies across the simulated world. Figure 3 shows examples of spreading patterns of populations for different settings of the diffusion speed parameter ω. Similar Turing patterns have been observed in the context of patterns of spreading of animals. In the case of human populations it is less clear the evidence for such patterns. However during fast migratory periods such patterns might have emerged, for example consider the cases of two (or more) Hungarian and Bulgarian states in the middle ages, one in the area of the current countries and another (possibly more than one) in the area what is today Eastern Ukraine and South Russia (the latter fell apart during the Mongol invasion) (Tomory et al, 2007; Johanson, 2000).

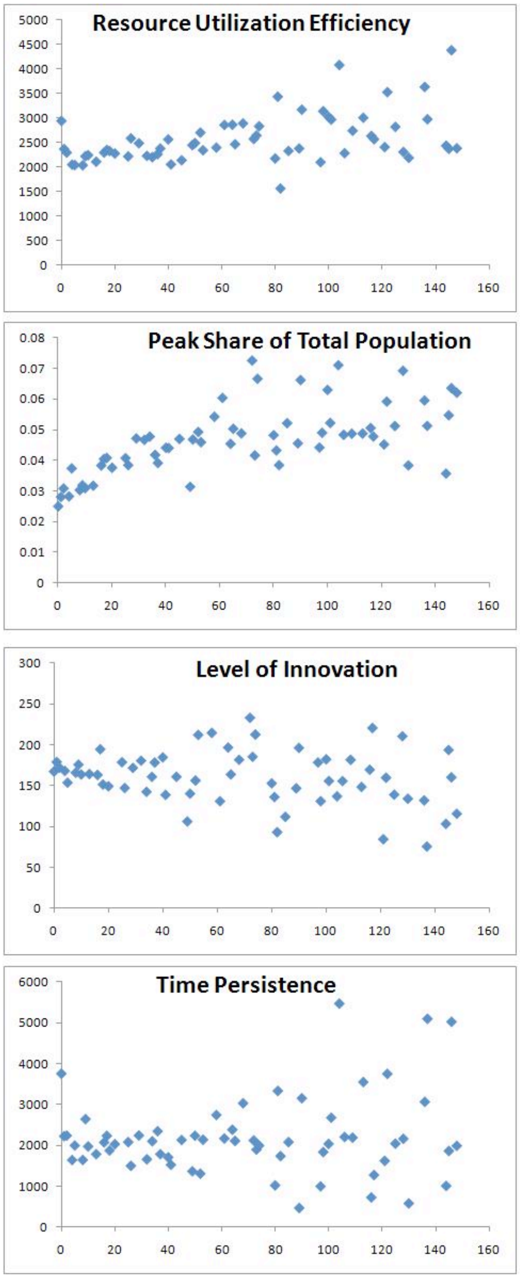


Figure 4. The relationships between the ridge distance of the location of origin of populations and the resource utilization efficiency, peak share of total population, level of innovation, and time persistence of simulated populations. The horizontal axis in all cases is the ridge distance of the population origin.

We analyzed the relationship between the distance of the location of origin of populations from the closest simulated mountain ridge (i.e. ridge distance) and the measured population performance indicators. For this purpose we pooled data from 20 simulations and calculated the average population persistence time, peak population share, resource utilization efficiency, and innovation value for each value of the ridge distance for which we found with an origin location having this value of ridge distance. The further analysis was done by considering only those ridge distance values for which we found at least 12 populations across all simulations.

We found that there is significant positive correlation ($p<0.05$) between the ridge distance and the average resource utilization efficiency ($c=0.4926$) and the peak share of the total population ($c=0.6263$). We found significantly negative correlation ($c= – 0.2793; p<0.05$) between the ridge distance and the level of innovation (i.e. difference in resource utilization efficiency compared to the immediate ancestor population). Finally we found no significant correlation between the ridge distance and the persistence time of the population. The data is shown in Figure 4 – error bars are not shown to avoid cluttering of the figures, the error bars are in general around 10 – 30% of the average values.

These results indicate that Hypothesis 1 is confirmed in the sense that being close to simulated mountain ridges has a positive impact on the level of innovation in terms of resource utilization efficiency by newly emerging populations. The data does not show whether this effect is due to the protection against invasion by other populations by the mountain ridges, which are expected to slow down the spreading of populations. The results also show that populations with higher resource utilization efficiency and higher peak share of the total population emerge further away from mountain ridges.

To analyze the relationship between land fertility of the location of origin of populations and the population performance indicators we pooled the data from 20 simulations. We calculated the average performance indicators for all values of land fertility that we found and considered for further analysis those averages that were calculated from at least 12 instances of populations. The data are presented in Figure 5.

We found that there is a significant ($p<0.05$) positive correlation between the harshness value (the opposite of land fertility) of the location of origin and the level of innovation of populations ($c=0.4146$ – note however the wavy nature of the relationship) and a significant negative correlation between the harshness value and the population's peak share of total population ($c= – 0.9275$). These two results confirm the validity of Hypotheses 2 and 3. The first result shows that indeed, the likelihood of high positive innovation increases with the harshness (low land fertility) of the location of origin of the population among populations that become sufficiently dominant. The second result shows that higher land fertility (lower harshness) of the location of origin of the population implies higher chance for the population to become highly dominant. This analysis of the data does not provide the definite explanation for these results, but the data allows further analysis, which may provide support for the hypothetical explanations that we provided in the previous section.

In terms of the relationship of the harness of the location of origin and the resource utilization efficiency and the time persistence of the population we found an unexpected relationship with three peaks at a two lower and a higher mid-level harshness value. The triple peak nature of the relationship also applies to the level of innovation as well.

To understand better the finding about the relationship of the harshness of the location of origin and the resource utilization efficiency and the time persistence of the

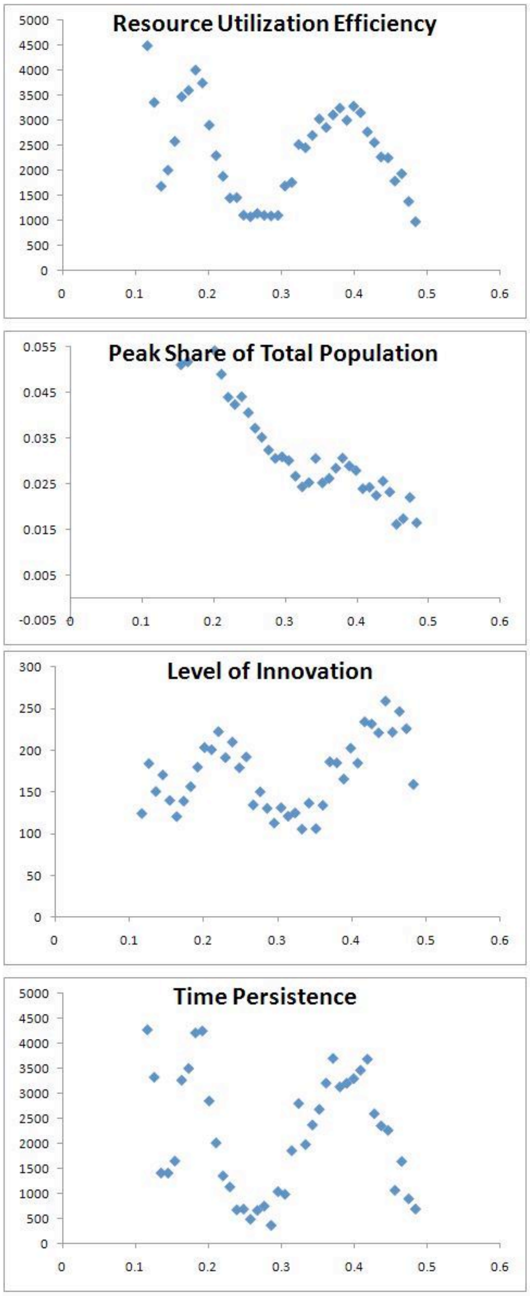


Figure 5. The relationships between the harshness of the location of origin of populations and the resource utilization efficiency, peak share of total population, level of innovation, and time persistence of simulated populations. The horizontal axis in all cases is the harshness of the population origin.

population we varied the values of the parameters γ and α, which characterize the default population growth and the level of competition among populations residing at the same spatial location. We considered γ values in the range of *0.8* to *1.7*, and α values in the range of *6* to *12*. We found that the behavior of these two relationships shows a wave-like nature with a number of peaks in general – see Figures 6 and 7. Note that these waves are not spatial waves in the simulated world,

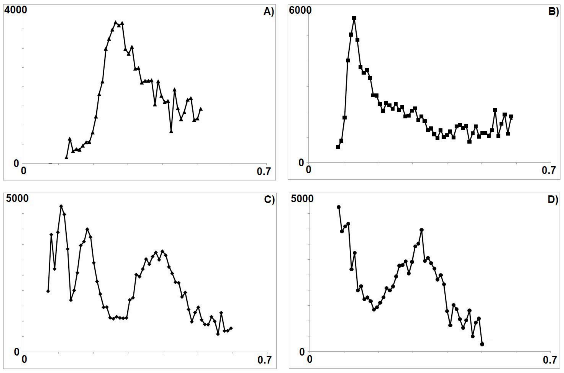


Figure 6. The average resource utilization efficiency of sufficiently dominant populations as function of the harshness of the location of origin – A) α=*6*; B) α=*8*; C) α=*10*; D) α=*12*; the lines connect the data points.

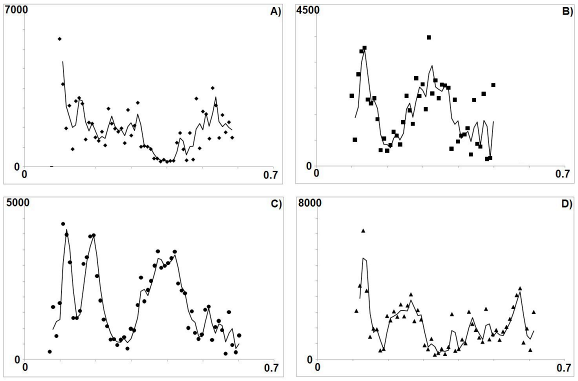


Figure 7. The average time persistence of sufficiently dominant populations as function of the harshness of the location of origin – A) γ=*0.8*; B) γ=*1.1*; C) γ=*1.4*; D) γ=*1.7*; the lines are the moving averages.

but waves depending on the harshness of the location of origin of the population.

Considering equation (1) that describes the general process of population spreading, the result about the wave-like dependence of the resource utilization efficiency and time persistence on the harshness of the location of origin suggests that reaction term of equation (1) – *F(u,x,t)*, which depends on the land fertility of the location, plays an important role in determining which locations are more likely to give birth to the longest existing and most resource efficient populations. These locations are consequently relatively over-represented among the locations of origins of populations that are sufficiently dominant. A possible explanation is that there is some interference between the wave like nature of population spreading (see the Turing patterns above in Figure 3) and the land fertility dependent reaction term of the population spreading equation, which constrains location of origin of the most resource efficient and longest existing populations to certain bands of land fertility values and locations which have such land fertility.

The implication of this result is that it is possible that for real human, animal, and plant populations as well, the locations of origins of most populations may not be the most fertile and supportive lands, but rather places with intermediate level of land fertility (or environmental supportiveness in general in the case of animals and plants), which match in some sense the wavelength of the waves of spreading of these populations.

Conclusions

In this paper we investigate the role of environmental factors in the emergence of cultural and technical innovations in human populations through simulation based analysis. We did not consider the possibility of copying of innovations by co-located populations and neither the possible impact of available excess resources. We used a relatively simple simulated world with simulated mountain ridges and low fertility land areas to analyze the impact of these factors on the successful spreading and survival of simulated human populations. The results show that both environmental factors matter for the emergence of successful populations.

Our results show that in the simulated world being close to mountain ridges correlates with the development of innovations with significant positive effect on the resource utilization efficiency in populations originating from such spatial locations. This matches well with the observation that several highland areas with mountain protected valleys facilitate the existence of many distinct human populations, each developing different cultural and technical innovations (Bulayeva et al, 2003; Reesink et al, 2009). Our simulations suggest that such areas are likely to lead to the emergence of cultural and technical innovations that give a significant competitive advantage to a local human population in comparison with neighboring populations and these populations are likely to spread out form these valleys and may become dominant populations in larger areas.

The results show that populations originating from locations with low land fertility are likely to develop significant positive innovations. The underlying reason may be that other populations originating from such areas that do not develop such significant positive cultural and technical innovations simply do not survive. The implication of this finding is that it can be expected that human populations with the most beneficial cultural and technical innovations may originate from areas that are particularly challenging environments.

Finally, we found that the most dominant simulated human populations originate from locations with high land fertility within the simulated worlds. This finding matches well with the evidence that shows that historically many of the most successful human populations get established in high land fertility areas (e.g. the fertile alluvial valleys along large rivers such as the Nile, Tigris, Euphrates, Huang He, Yangtze) (Fukuyama, 2014).

The three key findings in combination suggest that possibly the most successful human populations may have roots in harsh lands, possibly protected by mountain ridges, where their ancestors developed key cultural and technical innovations. The descendants of these ancestral populations may become really successful following migration and arrival to fertile land areas where they can expand and take full benefit of their competitive advantage relative to other populations, which is due to their cultural and technical innovations.

References

Barraclough, T.G., Vogler, A.P., Harvey, P.H. (1998). Revealing factors that promote speciation. Philosophical Transactions of the Royal Society of London B, 353: 241-249.

Bouckaert, R., Lemet, P., Dunn, M., Greenhill, S.J., Alekseyenko, A.V., Drummond, A.J., Gray, R.D., Suchard, M.A., Atkinson, Q.D. (2012). Mapping the origins and expansion of the Indo-European language family. *Science*, 337: 957-960.

Bulayeva, K., Jorde, L.B., Ostler, C., Watkins, S., Bulayev, O., Harpending, H. (2003). Genetics and population history of Caucasus populations. Human Biology, 75(6): 837-853.

Cheng, H., Yao, N., Huang, Z.-G., Park, J., Do, Y., Lai, Y.-C. (2014). Mesoscopic interactions and species coexistence in evolutionary game dynamics of cyclic competitions. Scientific Reports, 4: 7486.

Chiaroni, J., King, R.J., Underhill, P.A. (2008). Correlation of annual precipitation with human Y-chromosome diversity and the emergence of Neolithic agricultural and pastoral economies in the Fertile Crescent. *Antiquity*, 82: 281-289.

Delre, S.A., Jager, W., Bijmolt, T.H.A., Janssen, M.A. (2010). Will it spread of not? The effects of social influences and network topology on innovation diffusion. Journal of Product Innovation Management, 27: 267-282.

Der Sarkissian, C. et al (2013). Ancient DNA reveals prehistoric gene-flow from Siberia in the complex human population history of North East Europe. PLoS Genetics, 9(2): e1003296.

Diamond, J. (1997). *Guns, Germs, and Steel: The Fates of Human Societies*. Norton, New York.

Diamond, J., Bellwood, P. (2003). Farmers and their languages: The first expansions. Science, 300:597-603.

Fukuyama, F. (2014). *Political Order and Political Decay*. Profile Books, London.

Garcia-Ramos, G., Rodriguez, D. (2001). Evolutionary speed of species invasions. Evolution, 56(4): 661-668.

Jeltsch, F., Muller, M.S., Grimm, V., Wissel, C., Brandl, R. (1997). Pattern formation triggered by rare events: lessons from the spread of rabies. Proceedings of the Royal Society of London B, 264: 495-503.

Johanson, L. (2000). Linguistic convergence in the Volga area. Studies in Slavic and General Linguistics, 28: 165-178.

Lambin, X., Elston, D.A., Petty, S.J., MacKinnon, J.L. (1998). Spatial asynchrony and periodic travelling waves in cyclic populations of field voles. Proceedings of the Royal Society of London B, 265: 1491-1496.

Macy, M.W., Willer, R. (2002). From factors to actors: Computational sociology and agent-based modeling. Annual Review of Sociology, 28: 143-166.

Mahajan, V., Muller, E. (1979). Innovation diffusion and new product growth models in marketing. Journal of Marketing, 43(4): 55-68.

Mellars, P. (2006). Why did modern human populations disperse from Africa ca. 60,000 years ago? A new model. *PNAS*, 103(25):9381-9386.

Ouyang, Q., Li, R., Li, G., Swinney, H.L. (1995). Dependence of Turing pattern wavelength on diffusion rate. Journal of Chemical Physics, 102(6): 2551-2555.

Petrovskii, S.V., Morozov, A.Y., Venturino, E. (2002). Allee effect makes possible patchy invasion in a predator-prey system. Ecology Letters, 5: 345-352.

Reesink, G., Singer, R., Dunn, M. (2009). Explaining the linguistic diversity o Sahul using population models. PLoS Biology, 7(11): e1000241.

Tomory, G., Csanyi, B., Bogacsi-Szabo, E., Kalmar, T., Czibula, A., Csosz, A., Priskin, K., Mende, B., Lango, P., Downes, C.S., Rasko, I. (2007). Comparison of maternal lineage and biogeographic analyses of ancient and modern Hungarian populations. American Journal of Physical Anthropology, 134: 345-368.

Warren, D.L., Glor, R.E., Turelli, M. (2008). Environmental niche equivalency versus conservatism: quantitative approaches to niche evolution. Evolution, 62(11): 2868-2883.

Where Does (Co)evolution Lead to?

Jan Paredis

Department of Knowledge Engineering, Maastricht University,
P.O. Box 616, 6200 MD Maastricht, The Netherlands,
j.paredis@maastrichtuniversity.nl

Abstract

This paper investigates the dynamics of a simple coevolutionary system. It consists of a predator-prey system in which one population maximizes its distance to the members of the other population, while the second population tries to minimize the distance to the first population. This results in a coevolutionary pursuer-evader (PE) system whose dynamics can easily be studied.

Next, a simple genotype-phenotype mapping is added to the system. This mapping - as well as other sources of increased selection - push the system towards regions of maximum adaptability (ROMAs). These ROMAs are a generalization of the concept ”evolution to the edge of chaos”.

Introduction

Three concepts are central in this paper: evolution to the edge of chaos, Genotype Phenotype Mappings (GPMs), and coevolution. Each of these concepts is briefly introduced here.

The knowledge that evolution leads to the *edge of chaos* dates back to the late 80ies, early 90ies, see, for example Packard (1988), Kauffman and Johnsen (1991), and Langton (1990). A more critical account of evolution to the edge of chaos can be found in Mitchell et al. (1994). Paredis (1997) suggested why evolution leads to the edge of chaos:

> At each transition between order and chaos there is a high density of different dynamical behaviors, in a volatile environment, it is better to be prepared for change!

However, this hypothesis has not yet been proven empirically or theoretically. Or, as Wang et al. (2011) discuss:

> Much has been speculated on the possibility that gene regulatory and other biological networks function in (or evolve to) the critical regime (Gershenson, 2004).

The current paper investigates the evolutionary dynamics of *Genotype to Phenotype Mappings (GPMs)* in a coevolutionary model. Due to the rather unexpected small number of genes in the human genome, the interaction of these genes during morphogenesis has become an important research topic. It is the GPM which implements morphogenesis. Hence, its importance. In nature this GPM is complex. Or, as Benfey and Mitchell-Olds (2008) describe it:

> The difficulty in mapping genotype to phenotype can be traced to several causes, including inadequate description of phenotypes, too little data on genotypes, and the underlying complexity of the networks that regulate cellular functions.

More specific, the concept of evolvability has been introduced. It states that the GPM is under evolutionary pressure as well. It is assumed that these mappings are evolved in order to make the GPM robust and successful in various different (changing) environments. Or, as Benfey and Mitchel-Olds Pigliucci (2010) discuss:

> This re-thinking of the genotype-phenotype relationship and its consequences in terms of the related concepts of robustness, modularity and evolvability (Wagner, 2005) are part of an emerging Extended Synthesis in evolutionary biology

Wagner and Altenberg (1996) put forward the hypothesis that GPMs are under genetic control and that evolutionary algorithms (EAs) can be used to investigate this. One of the advantages of such an approach is that experiments can be done that are impossible in nature (e.g. because of the lack of control over system parameters). The research proposed here is an instantiation of their proposal in a simple artificial coevolutionary context. The aim of the current research is not to develop biologically plausible models. Nature is far too complex for that. Here, a simple model is used to study the dynamics of GPMs.

Besides GPMs, this paper studies *coevolution*. After the seminal work of Hillis (1990) on predator-prey coevolution for optimization, quite a considerable amount of research has been done on the use of predator-prey coevolution. An overview of this research can be found in (Paredis, 2000). Basically, this research shows how the arms race resulting from predator-prey interactions can be used in order to obtain efficient optimization. Far less research has been devoted to the study of the dynamics of these coevolutionary algorithms. De Jong (1999) proposed this research area for traditional (single population) genetic algorithms. What makes coevolutionary algorithms different is that they have to deal with dynamic environments: the populations change all the time as a response to the changes in the other population. An overview on dynamic worlds can be found in Dempsey et al. (2009). Such a dynamic environment provides a rich basis for the dynamics.

The definition of evolvability is rather varied, or as Pigliucci (2008) expresses it mildly:

> The concept of evolvability has many different definitions, generating some conceptual confusion, or at least pluralism.

Dempsey et al. (2009) define evolvability differently (than above) and link it to dynamic worlds as follows:

> Evolvability is the potential to incrementally increase the population fitness without first experiencing a decrease at it escapes a local optimum. Where the environment is dynamic a high level of evolvability allows the population to more easily track a moving optimum.

In the current paper, the Coevolutionary Genetic Algorithm (CGA), introduced in Paredis (1994), is used. In the past the CGA has mainly been used as a tool for optimalisation, see e.g. Paredis (1995). Now, the dynamics of the CGA is studied. The coevolutionary interactions in nature are often complex. The goal of this paper consists of the design of a SIMPLE coevolutionary application and GPM which - despite their simplicity - still exhibits realistic, complex dynamics.

The structure of this article is as follows. First, the standard CGA is described. Next, the simple application which results in Pursuer Evader (PE) dynamics is described. Next, the genotype-phenotype mapping is introduced. The fifth section describes the empirical results associated with the GPM. Next, related research is discussed. The seventh section discusses the model used and its relation to the real world, followed by a road map for future research. Finally, conclusions are given. Note that a short version of the current paper is published in Paredis (2014). This short paper does not include any empirical results.

A Coevolutionary Genetic Algorithm

Here, the basic CGA is described, as a first step it creates, two populations (called pop1 and pop2). Typically, the individuals in these initial populations are (uniformly) randomly generated. Next, the fitness of these individuals is calculated. This fitness depends on the particular application, but it is the result of a number - here 10 - of ENCOUNTERS of an individual with individuals of the other population. These encounters result in a pay-off which is stored in the history of the individual. The actual fitness is the average of these (10) history elements. Because these encounters represent predator-prey interactions, success for one individual (in an encounter) is failure for the other one. Hence, the value of an encounter is stored in the history of one individual involved in the encounter. The other individual stores the negative of this value in its history. Once all initial fitnesses are calculated, both populations are sorted on fitness: the individual with the highest fitness on top the least fit one at the bottom.

Next, the main *cycle* of a CGA is executed. The pseudocode of this cycle is given below. First, 20 encounters are executed between SELECTed individuals. This selection is linearly biased towards highly ranked individuals: similar to GENITOR (Whitley et al., 1989) the top individual is 1.5 times more likely to be selected than the median individual. Next, the pay-off of this encounter is calculated and stored in the history, removing the payoff of the least recent encounter from the history. Hence, the history is implemented as a queue. Finally, the fitness (the average of the history) of both individuals involved in the encounter is re-calculated. Possibly, this changes the ranking of the individual in its population. Note that the predator prey interaction results in a negative pay-off for the individual of the second population.

```
DO 20 TIMES
   ind1 := SELECT(pop1)
   ind2 := SELECT(pop2)
   payoff := ENCOUNTER(ind1,ind2)
   UPDATE-HISTORY-AND-FITNESS(ind1,payoff)
   UPDATE-HISTORY-AND-FITNESS(ind2,-payoff)
ENDDO

p1 := SELECT(pop1)         ; pop1 parent1
p2 := SELECT(pop1)         ; pop1 parent2
child := MUTATE-CROSSOVER(p1,p2)
f := FITNESS(child)
INSERT(child,f,pop1)
p1 := SELECT(pop2)         ; pop2  parent1
p2 := SELECT(pop2)         ; pop2  parent2
child := MUTATE-CROSSOVER(p1,p2)
f := FITNESS(child)
INSERT(child,f,pop2)
```

After these 20 encounters the CGA produces one offspring for each population: it SELECTs two parents. A new individual is generated from these parents through the application of MUTATion (probability of mutating a gene is 0.1) and (uniform) CROSSOVER. The fitness is calculated by executing 10 encounters between the new individual and SELECTed members of the other population (again using the negative payoff for individuals which belong to the second population). In case this fitness is higher than the fitness of the bottom individual then the new individual is placed in the population at its appropriate rank. All individuals with a lower fitness go one position down and the bottom individual is deleted. This basic cycle is repeated a large number of times (e.g. 20000 cycles). In the current paper, all parameter settings and genetic operators are identical to those described in Paredis (1995) unless mentioned otherwise.

Pursuer-Evader Dynamics

In this particular application, each individual consists of two genes: real numbers in the interval [0,1]. The pay-off of an encounter between two individuals consists of the cartesian distance between the two pairs of genes. The first population maximizes the distance to the individuals of the other population. The negative payoff of the members of the second population results in a minimization of the distance to the individuals of the first population. This because in both populations fitness is maximized.

Each individual can be represented as one point on the plane [0,1] x [0,1]. Furthermore, in order to allow for an unbounded evolution, this plane is considered to be a torus. Hence, the distance is the minimum of the two possible distances (one crossing (an) ”edge(s)”). Furthermore, mutation can cross the ”edges” as easy as it can move in the plane. Or, in other words, 0.95 is equally likely to be mutated into, for example, 0.085 or 0.05. Finally, a standard uniform crossover is used: new offspring receives each gene from one of its parents randomly and independently.

The dynamics of this application is fairly simple. The initial (random) populations are scattered randomly over the plane. In the first experiment described below equal population sizes consisting of 50 individuals are used. Fairly soon (typically in less than thousand cycles) during evolution two clusters appear (one for each population) where one cluster chases (pursuer) the other (evader). From time to time different behavior is observed. Sometimes the pursuers catch up on the evaders. At this moment the cluster of evaders breaks up. Most of the time the evader cluster breaks up in two or four sub clusters, which are located symmetrically with respect to the pursuers. These sub clusters virtually immobilize the pursuers while the evader sub clusters move radially and finally become one cluster again. Due to sampling errors and finite population sizes the evaders cluster (i.e. unite) again before the sub clusters have gone all the way. Once the evaders are clustered again, the ”standard” pursuing of two clusters continues.

When the two populations have different population sizes then their respective speed changes. This is because at each cycle both populations reproduce once. Hence, the smaller population evolves the fastest, i.e. moves faster on the plane. In case the pursuer population is smaller, the pursuers regularly catch up with the evaders. When this happens the evaders split up, again immobilizing the pursuers until the evaders form one cluster again. Then the chase resumes. In the other case, the evader population is the smallest population. Here, the evader population successfully keeps ahead of the pursuer population. Occasionally, the evaders even have to slow down in order not to get too close to the pursuers (remember the world consists of a torus).

Introducing a Genotype-Phenotype Mapping

In this section, the PE-model is extended with a simple GPM in order to study its contribution to the dynamics. To add a GPM two real numbers from the interval [0,1] are added to the gene string of each evader and pursuer. This string then takes the following form: (x y r_1 r_2). The two first parameters are as before. The last two define the GPM for each of the two parameters (x and y) independently. Hence, the GPM is under evolutionary control.

Each time a new individual is born - this happens once per population and per cycle) the GPM functions as follows. The two iterative mappings, equations (1), and (2), are applied to x and y, respectively, a fixed number of times. This number is called ”pregtime”. Here, in all experiments, α is set to 0.001. After this, the individual’s fitness (through encounters, just as before) is calculated and if fit enough it is inserted in the population at the appropriate rank. Note that the GPM changes the original x and y genes. Or, more precisely, the genotype remains the same, the phenotype - which is originally a copy of the genotype - changes. Also the GPM considers the genes to live on a torus: if it becomes larger than 1, 1 is subtracted from it, x and y cannot become negative because α as well as r_1 and r_2 are all positive.

$$x_{n+1} = x_n + \alpha r_1 x_n \tag{1}$$

$$y_{n+1} = y_n + \alpha r_2 y_n \tag{2}$$

After birth, these iterative functions are applied once during each cycle of the CGA and that during a fixed number of cycles. This constant is called ”growtime”. Both processes are simple models of a PGM operating in two phases. The first phase represents development during pregnancy, during which no interaction with the world occurs. The second one represents further development after birth, i.e. growth, during which the phenotype interacts with the world. Here, pregnancy as well as growth are represented by small positive increments of the phenotype.

Empirical Results

Given the PE behavior of the standard CGA described before, the question now is: How do r_1 and r_2 evolve in various settings? In order to study this, 100 runs (of 20000 cycles) of the PE CGA with PGM were run. Then all r’s belonging to one population are printed on a [0,1] x [0,1] plane. Hence, these are the values of the r’s at the end of each of the 100 runs. In this paper, an experiment will be described by a quadruple: the first element is the size of the evader population, the second the size of the pursuer population, the third the pregtime, and finally, the growtime.

In the first case pregtime and growtime are both zero, the r’s play no (evolutionary) role and they will be distributed randomly over the plane. At the end of each run, the r_1 and r_2 of each evader or pursuer are plotted as a dot on a

[0,1]x[0,1] plane. Figure 1 shows the distribution of the r's of the evaders in the experiment represented by the quadruple 50-50-0-0: both population sizes are 50 and no GPM is used. Any pattern in this figure is due to random drift and the relatively small number of runs (100). Because of the symmetry between the evaders and pursuers (same poulation size, and both populations do not use a GPM), the r's of the pursuers show a similar distribution as the evaders.

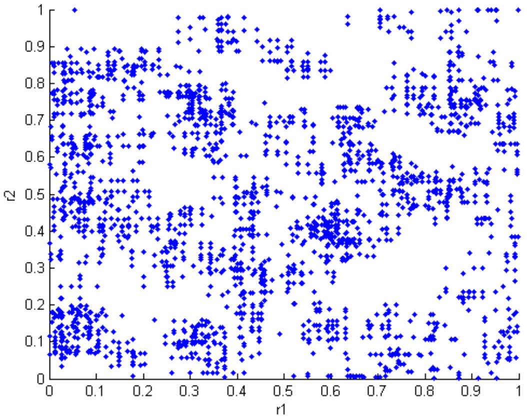


Figure 1: Distribution of r's of evaders for 50-50-0-0

In the next experiment - 50-50-80-20 - pregtime is set to 80 and growtime to 20. Or, in other words, directly after birth of an evader or pursuer the PGM is applied 80 times, during the following 20 cycles the PGM is applied one time per cycle on that individual. Figure 2 shows the distribution of the r's of the evaders. Clearly, there is structure now: the r's concentrate around the "edges". They even seem to concentrate most at the "corners". The explanation why this happens is extremely simple: at these places mutation results in the largest variation of change (i.e. change of movement of the x,y genes on the [0,1] x [0,1] torus). At these edges, it switches from maximal movement to zero movement and vice versa. This allows the individuals of each population to out-manoeuvre the individuals of the other population. The term *regions of maximum adaptability* (ROMAs) is used here to define these regions where the r's (and hence the GPM) evolve to. Actually, the term ROMA was coined in Paredis (2014). In terms of dynamic systems, these ROMAs are attractors and the density of the r's give an indication of their strength. Again, due to symmetry (same population size, and same amount of GPM), the distribution of the r's of the pursuers is similar to the distribution of the r's of the evaders.

Now, the influence of differences in population size is investigated, again pregtime is set to 80, and growtime is 20. In this experiment, the population size of the evaders is set to 20, the pursuers are still with 50. Figure 3 depicts the distribution of the r's of the evaders. Obviously, there are less dots now because the population size is smaller. Furthermore, there is no strong pressure towards the edges (or corners). This in comparison with the distribution of r's of the pursuers (figure 4). It can be seen that for the largest population (that is the pursuers) the pressure towards ROMAs is the strongest. The larger populations move slower because at each cycle one offspring is generated. Hence, it is more important, for their survival, that they have access (genetically and / or through the GPM) to a wide variety of behavior. Such that they can out-manoevre the smaller, faster population.

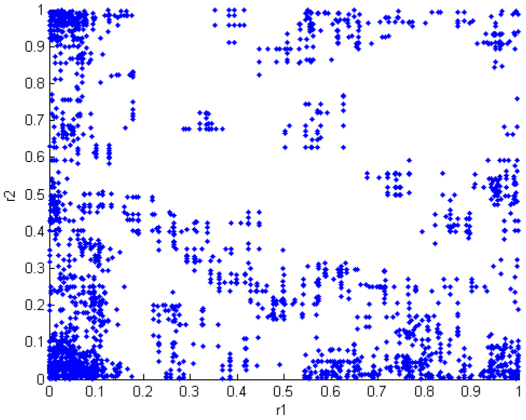


Figure 2: Distribution of r's of evaders for 50-50-80-20

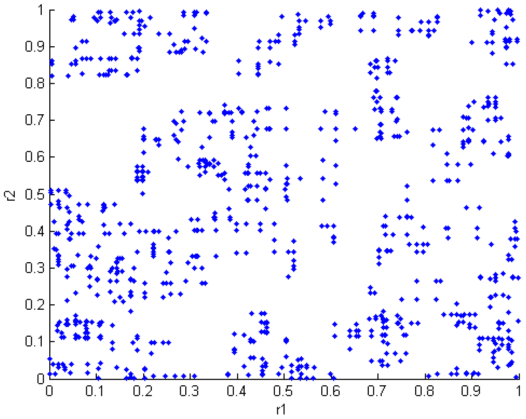


Figure 3: Distribution of r's of evaders for 20-50-80-20

The same phenomenon is observed when the population of pursuers is smaller (20) than the evader population (50). In this case, the r's of the evaders are pushed more towards the ROMAs. Clearly, the effect of the GPM, pushing the r's to the ROMAs, is enforced by the increased pressure on the largest population.

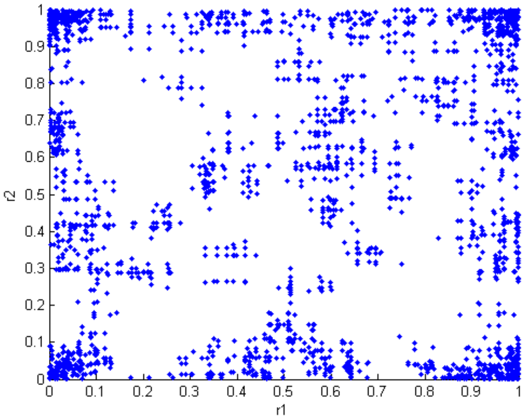


Figure 4: Distribution of r's of pursuers for 20-50-80-20

A final set of experiments studies the effect of changing the influence of the PGM and the effects of the interplay of its two phases: pregnancy and growth. When both, pregtime and growtime, are set to zero then the r's don't play any role. Hence, both r's are randomly distributed (as was the case in figure 1). When pregtime or growtime are set to hundred (and the other one remains zero) then the r's belonging to the largest population most clearly moves to the ROMAs. If both population sizes are equal both r's move to the ROMA in an equal way as was seen in figure 2. This phenomenon is most outspoken when pregtime is hundred. In this case, the influence of the PGM is largest: on each individual the PGM is applied 100 times. In the other case, it depends on the age, i.e. the number of cycles executed since its birth, the individual attains. In a dynamic world, the individuals are typically short lived. Figure 5 depicts the evolution of the average age of both populations during a typical run of 20-50-80-20 over 10000 cycles. As a matter of fact, the individuals belonging to the smaller population have the shortest life span. Hence, the effect of the GPM is most outspoken / guaranteed during the pregnancy. As a result, the pressure towards the ROMAs will be the strongest when pregtime is set to hundred (and growtime equals zero). Summarizing, pressure towards ROMAs is stronger for (comparatively) larger populations and with increasing role of the PGM. Or, yet in other words, as selection pressure increases the push towards ROMAs becomes stronger.

In fact, one can evolve more parameters of the GPM than only the r's, such as the α's. This way, the balance between evolution and the GPM can be evolved and studied. One step further is to evolve the constituent parts of the GPM (the pregtime and growtime). This provides insight into the interplay between these components under different circumstances.

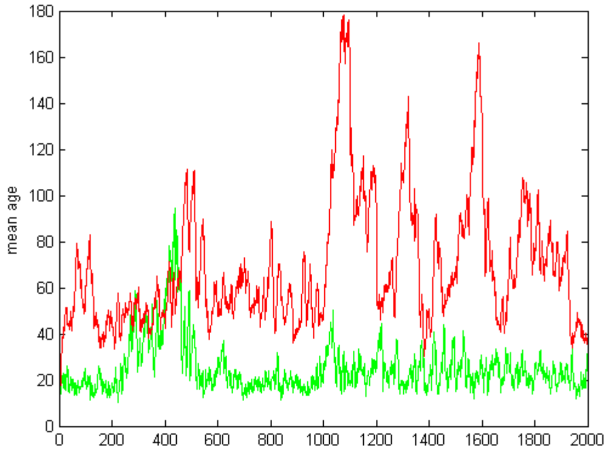


Figure 5: Evolution of the average age of both populations during a typical run of 20-50-80-20 over 10000 cycles. The x-axis represents the number of cycles (at each fifth cycle). The lower graph represents the average age of the evaders.

Related Research

Ebner et al. (2010) investigate the dynamics of coevolutionary interacting species as well. One important difference is the level at which the interactions are modelled. Ebner et al. model at the level of species, whereas the current paper models the interactions at the level of individuals.

Suzuki and Kaneko (1994) describe research closely related to the PE described here. They use a single population of birds. The fitness of a bird is determined by: 1) How hard it is for other birds to imitate its song, and 2) How good it is at imitating the song of the other birds. In fact, in that model, evolution leads towards the edge between chaos and a periodic window.

The research of Castro de Oliveira de Oliveira (2001) is particular relevant here. This work states that eternal transients are needed such that the system never reaches its final destiny. These can be found at the edge of chaos. Literally:

> Evolutionary systems stick to critical situations (the edge of chaos), because within all other possibilities (regular or chaotic regimes) they rapidly become trapped into low-dimensional attractors, loosing diversity. This trapping feature forbids the system to explore the whole set of available possibilities inside its higher-dimensional space of states.

The current paper provides a far simpler and more general explanation for evolution to the edge of chaos: in a dynamic world evolution leads towards ROMAs. In these regions a maximum repertoire of behaviors is easily accessible through the application of genetic operators and / or the GPM. This is exactly what happens near the edge of chaos: it is there that the most diverse behaviors are situated. Hence, ROMAs generalize this concept of evolution to the edge of chaos. Or, from a slightly different perspective, ROMAs are the regions where evolvability is high. In retrospect, the results presented here seem rather intuitive. However, extensive experimentation was needed to obtain these results and to understand them.

Discussion

The fact that a very simple model is used here, can give raise to questions like: How realistic is this model? How do the results carry-over to the real world? Are the results not mere artefacts of the model chosen?

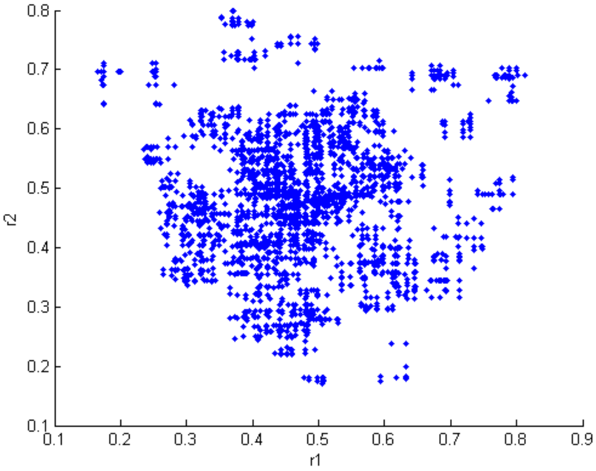


Figure 6: Distribution of r's of evaders for 50-50-80-20 using non-toroidal mutation of the r's

.

As described before, the fact that a toroid model is used here is to allow for infinite, ongoing evolution. Besides this, the toroidal model allows to have different behaviors near each other in the gene space. In the PE model used here, there is no real interaction (epistasis) between the genes of one individual. The interaction is with the individuals of the other population. In nature, epistasis (within and between populations) allows for small changes in genes to result in large effects (i.e. different behavior). Here, the toroidal model plays the role of epistasis: both create regions where small changes lead to large effects. Hence, in other words, epistasis gives rise to ROMAs.

Actually, experiments were done where mutation of the r's is no longer toroidal (but mutation of x and y still is). In these cases, different patterns of r's are formed: a cloud is formed with r_1 and r_2 roughly concentrated at the centre of the plane (see figure 6 for the distribution of the r's of the evaders of 50-50-80-20). As before when selection pressure is increased - through a larger population size (relative to the other population) or "more" GPM - the r's concentrate more at the centre of the plane (see figure 7 for 20-50-80-20).

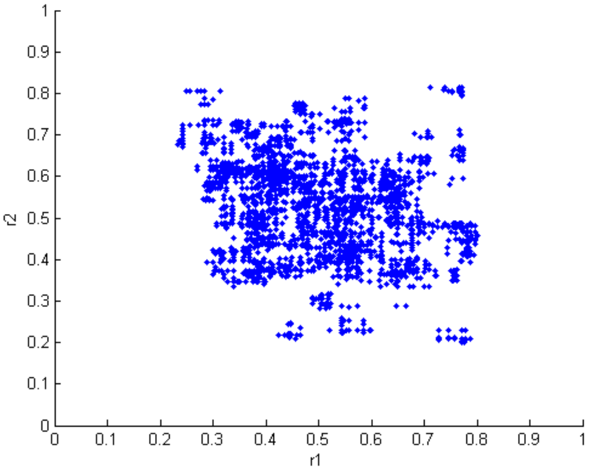


Figure 7: Distribution of r's of pursuers for 20-50-80-20 using non-toroidal mutation of the r's

.

These figures clearly indicate the impact of non-toroidal mutation versus the toroidal mutation used earlier. Whereas increased selection pressure pushes the r's to the edges and corners in case of toroidal mutation, non-toroidal mutaion results in a pressure towards the centre.

Summarizing, the toroidal nature of mutation, might seem artificial, but it compensates for the lack of epistasis in this simple PE model.

Future Research

As stated before, the PE model used here is extremely simple. This is because the goal was to take the simplest model to study interesting coevolutionary dynamics (including PGMs). These simplifications provide a road map for future research which allows to investigate the respective implications / contributions of these restrictions, and their relaxations, to the dynamics.

First of all, the PGM can be made more realistic. Now, it is a one to one mapping. It is known that typically PGMs are one to many mappings. This should further increase evolvability. Furthermore, the fact that the genotype space is similar to the phenotype space is unrealistic.

Another road is to use a different application instead of the pursuer evader system. The PE was used here because it would provide interesting and everlasting dynamics.

Also the fixed (and small) length of the individuals could be relaxed in order to see whether this has an effect on the dynamics.

Obviously, parameters could be set different for each population. Now all parameters, except the experiments with different population sizes, were kept identical. This could include parameters like reproduction rate, mutation rate, size of mutation ... . Here again, the effect of these changes on the dynamics can be investigated. As already mentioned before, the parameters of the GPM, such as pregtime and growtime, could be evolved as well.

A final remark is that although the application discussed here (PE) is spatial. The evolution, however, is not organized in a spatial way, i.e. a mixed model is used here. Clearly, a spatial setup in which selection and reproduction is local has been shown to be more natural and more efficient at optimizing Pagie and Hogeweg (1997). It is expected that the use of a CGA with a spatial component would have an important impact on (the complexity of) the dynamics.

Finally, it is good to note, that the original aim of the research reported here was to investigate the dynamics of the CGA described in Paredis (1995). That is why a non-spatial organisation is used. For the same reason, each population reproduces once per cycle. Clearly, this is not the case in nature. Nonetheless, the general principle discussed here - increased selection pressure providing a push towards ROMAs - is likely to carry over to nature. This because the basic processes of natural evolution are used here: variation, selection, and reproduction with inheritance.

Conclusion

This paper studies the dynamics of a coevolutionary (predator-prey) algorithm by means of a simple pursuer evader application on a torus. This application allows for easy visual inspection of the dynamics. In case both populations have the same size then each population rapidly clusters and the cluster of pursuers chases the cluster of evaders. Occasionally, the pursuers catch up with the evaders. These then split up in sub clusters symmetrically around the pursuers, immobilizing the latter for a while, until re-clustering appears.

Next, a simple genotype-phenotype mapping (GPM) was introduced. This mapping is under evolutionary control as well. As selection pressure increases the GPM evolves towards regions of maximal adaptability (ROMAs). From these regions the individual can easily change its behavior, and has access to a large repertoire of different behaviors. These ROMAs are a generalization of the concept ”evolution to the edge of chaos”. Or, in other words, the ROMAs are regions with high evolvability.

Acknowledgements

I am indebted to Ivo Bleylevens for getting me started with Matlab. Roos-Marie Bal proofread a draft of this paper. The comments of the ECAL-15 reviewers allowed to improve an earlier draft of this paper.

References

Benfey, P. N. and Mitchell-Olds, T. (2008). From genotype to phenotype: systems biology meets natural variation. *Science*, 320(5875):495–497.

De Jong, K. (1999). Evolving in a changing world. *Foundations of Intelligent Systems*, pages 512–519.

de Oliveira, P. M. C. (2001). Why do evolutionary systems stick to the edge of chaos. *Theory in Biosciences*, 120(1):1–19.

Dempsey, I., O’Neill, M., and Brabazon, A. (2009). *Foundations in Grammatical Evolution for Dynamic Environments*, volume 194. Springer.

Ebner, M., Watson, R. A., and Alexander, J. (2010). Coevolutionary dynamics of interacting species. In *Applications of Evolutionary Computation*, pages 1–10. Springer.

Gershenson, C. (2004). Updating schemes in random boolean networks: Do they really matter. In *Artificial Life IX Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems*, pages 238–243. MIT Press.

Hillis, W. D. (1990). Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, 42(1):228–234.

Kauffman, S. A. and Johnsen, S. (1991). Coevolution to the edge of chaos: Coupled fitness landscapes, poised states, and coevolutionary avalanches. *Journal of Theoretical Biology*, 149(4):467–505.

Langton, C. G. (1990). Computation at the edge of chaos: Phase transitions and emergent computation. *Physica D: Nonlinear Phenomena*, 42(1):12–37.

Mitchell, M., Crutchfield, J. P., and Hraber, P. T. (1994). Dynamics, computation, and the” edge of chaos”: A re-examination. In *Santa Fe Institute Studies in the Sciences of Complexity-Proceedings*, volume 19, pages 497–497. Addison-Wesley Publishing Co.

Packard, N. H. (1988). *Adaptation toward the edge of chaos*. University of Illinois at Urbana-Champaign, Center for Complex Systems Research.

Pagie, L. and Hogeweg, P. (1997). Evolutionary consequences of coevolving targets. *Evolutionary Computation*, 5(4):401–418.

Paredis, J. (1994). Steps towards co-evolutionary classification neural networks. In *Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pages 102–108.

Paredis, J. (1995). Coevolutionary computation. *Artificial life*, 2(4):355–375.

Paredis, J. (1997). Coevolving cellular automata: Be aware of the red queen. In *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 393–400.

Paredis, J. (2000). Evolutionary computation 2: Advanced algorithms and operators, chapter coevolutionary algorithms.

Paredis, J. (2014). (co)evolution leads towards romas. In *Proceedings of the European Conference on Artificial Intelligence 2014 (ECAI-14), doi: 10.3233/978-1-61499-419-0-1079*, pages 1079–1080. IOS Press.

Pigliucci, M. (2008). Is evolvability evolvable? *Nature Reviews Genetics*, 9(1):75–82.

Pigliucci, M. (2010). Genotype–phenotype mapping and the end of the genes as blueprintmetaphor. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 365(1540):557–566.

Suzuki, J. and Kaneko, K. (1994). Imitation games. *Physica D: Nonlinear Phenomena*, 75(1):328–342.

Wagner, A. (2005). *Robustness and evolvability in living systems*. Princeton University Press Princeton.

Wagner, G. P. and Altenberg, L. (1996). Perspective: Complex adaptations and the evolution of evolvability. *Evolution*, pages 967–976.

Wang, X. R., Lizier, J. T., and Prokopenko, M. (2011). Fisher information at the edge of chaos in random boolean networks. *Artificial Life*, 17(4):315–329.

Whitley, D. et al. (1989). The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In *Proceedings of the third international conference on Genetic algorithms*, volume 1, pages 116–121.

A New View of Protocell Metabolism

Ben Shirt-Ediss[1,2], Ricard Solé[1,3] and Kepa Ruiz-Mirazo[2,4]

[1]ICREA-Complex Systems Lab, Institut de Biologia Evolutiva, CSIC-UPF, Barcelona, Spain
[2]Logic and Philosophy of Science Department, University of The Basque Country, San Sebastián, Spain
[3]Santa Fe Institute, 1399 Hyde Park Road, Santa Fe, New Mexico, USA
[4]Biophysics Unit (CSIC, UPV/EHU), University of The Basque Country, Bilbao (Leioa), Spain
ben@shirt-ediss.me

Chemical reaction networks can exhibit interesting non-linear dynamics in reaction contexts with variable solvent volume (Pawłowski and Zielenkiewicz, 2004; Lizana et al., 2008), although this possibility has received little attention in the literature. Lipid vesicles, the standard chassis for many prebiotic protocell models, are compartments which enclose a water pool of variable volume. The semi-permeable nature of the lipid bilayer membrane facilitates, by the mechanism of osmosis, a flow of water entering or leaving the internal pool of the vesicle whenever there is a disequilibrium in the total solute concentration inside and outside (Oglecka et al., 2012). Therefore, the volume of a vesicle is not only variable, but is also determined as a function of the reaction dynamics ongoing inside the vesicle.

This contribution reviews and extends the findings of a new theoretical study (Shirt-Ediss et al., 2015) investigating novel non-linear behaviour which can arise when simple chemistry, that by itself is uninteresting, is encapsulated inside model lipid vesicles with osmotically-changing solvent volume (Figure 1). These findings could be relevant to the origins of life, at a stage when vesicles moved away from equilibrium and turned into the first chemical nano-reactors.

In particular, a new principle called *osmotic coupling* is to be put forward. Osmotic coupling refers to the potential of osmosis to *indirectly* couple two or more chemically-independent reaction networks operating inside a lipid vesicle. The coupling results because, despite not having any chemical species in common, the reaction networks happen to share the same variable volume reaction space, which their chemical concentrations jointly determine the size of. In this way, a larger chemical system with potentially very complicated dynamics may be created from simpler pieces.

This view of proto-metabolism, as a series of indirectly coupled but otherwise chemically inert subsystems, represents an interesting departure from the traditional dogma of proto-metabolism as one single connected set of chemical transformations (e.g. as in Ganti's Chemoton). Furthermore, it could be more realistic, considering that prebiotic vesicles would have assembled in diverse chemical mixtures hosting potentially unrelated reaction processes.

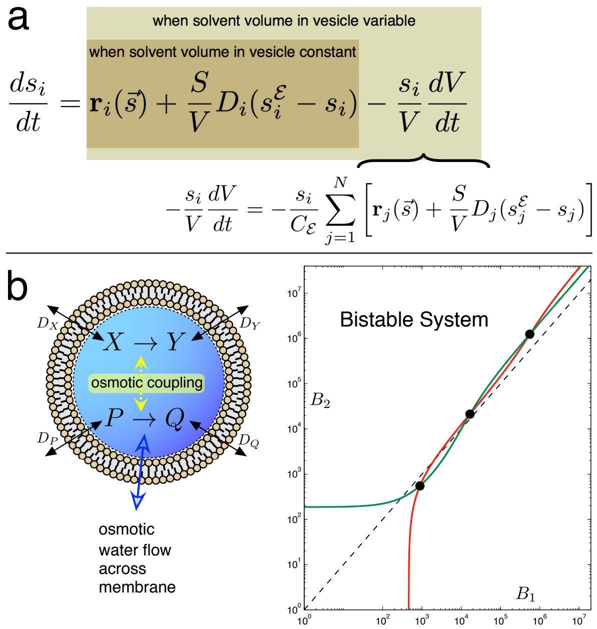

a
when solvent volume in vesicle variable
when solvent volume in vesicle constant
$\frac{ds_i}{dt} = \mathbf{r}_i(\vec{s}) + \frac{S}{V} D_i(s_i^{\mathcal{E}} - s_i) - \frac{s_i}{V}\frac{dV}{dt}$
$-\frac{s_i}{V}\frac{dV}{dt} = -\frac{s_i}{C_{\mathcal{E}}} \sum_{j=1}^{N} \left[\mathbf{r}_j(\vec{s}) + \frac{S}{V} D_j(s_j^{\mathcal{E}} - s_j) \right]$
b
D_X
D_Y
$X \to Y$
osmotic coupling
$P \to Q$
D_P
D_Q
osmotic water flow across membrane
Bistable System
B_2
B_1
10^0
10^1
10^2
10^3
10^4
10^5
10^6
10^7

Figure 1: Changing solvent volume inside a vesicle **(a)** theoretically introduces more non-linear terms to the concentration dynamics of each internal species and **(b)** can give rise to the emergence of new steady states, via osmotic coupling.

References

Lizana, L., Bauer, B., and Orwar, O. (2008). Controlling the rates of biochemical reactions and signaling networks by shape and volume changes. *Proc Natl Acad Sci U S A*, 105(11):4099–4104.

Oglecka, K., Sanborn, J., Parikh, A. N., and Kraut, R. S. (2012). Osmotic gradients induce bio-reminiscent morphological transformations in giant unilamellar vesicles. *Front Physiol*, 3(120):1–11.

Pawłowski, P. H. and Zielenkiewicz, P. (2004). Biochemical kinetics in changing volumes. *Acta Biochim Pol*, 51(1):231–243.

Shirt-Ediss, B., Solé, R. V., and Ruiz-Mirazo, K. (2015). Emergent chemical behavior in variable-volume protocells. *Life (Basel)*, 5:181–211.

Ribosome synthesis and construction of a minimal cell using a cell-free expression platform

Filippo Caschera, Michael C. Jewett

Department of Chemical and Biochemical Engineering,
Northwestern University, Evanston, 60208 Illinois,
United States of America

filippocaschera@gmail.com

Abstract

The creation of wet artificial life in the laboratory is a non-trivial challenge for biologists, chemists, and computer scientists (1-4). Such a challenge revolves around the modular integration of complex reactions networks to obtain functional biochemical units able of self-replication, self-reproduction, spatial-temporal control and ultimately open-ended evolution, e.g. minimal and artificial cells (1,5-8).

As a step towards building minimal cells, we have developed a cell-free expression system for bacterial ribosome synthesis named iSAT: integrated Synthesis, Assembly and Translation for *in vitro* construction of *Escherichia coli* ribosomes (9). The ribosomal RNA, transcribed from its natural operon, self-assembles with ribosomal proteins added to the reaction mixture. Afterwards, *in vitro* built synthetic ribosomes translate a reporter gene (10,11). Such system is important to design ribosome with new functions, and for the bottom-up construction of a minimal cell.

Ribosome cell-free synthesis is an essential process for building a minimal cell that can maintain itself (1). Indeed, regeneration of encoded DNA molecular machineries, through a compartmentalized reaction network, will be necessary to ensure gene expression after cycles of self-reproduction (12,13).

In this work, we have sought to improve the efficiency of the iSAT reaction to achieve the break-even milestone of ribosomes that are capable of constructing ribosomes (7,434 peptide bonds are needed to make a complete set of r-proteins). To do this, we prepared and optimized the iSAT reaction system using a robot for liquid handling. The open nature of cell-free expression platforms enables precise settings of each component level for an optimal system's configuration. Previously, high-throughput screening and machine learning have been used for the optimization of a cell-free protein synthesis and a liposomal drug formulation (14-16).

Here, I will present the optimization of *in vitro* ribosome construction using a cell-free expression system, and I will introduce future directions of the project. In particular, I will describe biochemical experimental spaces underlying the cell-free ribosome synthesis. I will show results on the optimization using a liquid handling robot for high-throughput experimentation. Our cell-free protein synthesis platform is the only one enabling *in vitro* ribosome construction, which is relevant to the synthesis of a minimal cell.

Adding effective energy regeneration modules is also important for minimal cell projects. Therefore, I will also present data highlighting the ability to regenerate ATP with a non-phosphorylated energy substrate with the iSAT system. Recently, I developed a novel metabolic scheme for a minimal cell (17). The system is based on the catabolism of polysaccharides and/or a polyphosphate (18-20) to regenerate ATP. Proteins are synthesized using a custom-made amino acids mixture (20).

The system is improved by overcoming a fundamental limitation: the efficient recycling of the orthophosphate (iP), which is the by-product of protein synthesis. As a result, ATP (adenosine triphosphate) is kept at steady state and available for *in vitro* transcription and translation (19). Currently, it represents the most powerful *in vitro* protein synthesis systems (18).

One important feature of a minimal cell is the compartment or container, which more than physically interlock components and sub-systems, confers the necessary genotype-phenotype linkage for evolution (21). The compartment of the minimal cell is based on liposomes (1-3,22), and evolutionary dynamics such as fusion and division, are important for resource feeding and selection respectively (6,8,12).

In summary, the work described is designed to lay the foundation for the construction of a synthetic replicating entity by building up synthetic biological unit operations (*e.g.* cell-free synthesis of constituent parts) and fine-tune the starting blueprint. Indeed, through the bottom-up synthesis of a minimal cell, we are building and understanding complex biological systems.

References

Journal Article

1.Caschera, F. and V. Noireaux. 2014. Integration of biological parts toward the synthesis of a minimal cell. *Current Opinion in Chemical Biology*, 22C:85-91.

2.Noireaux, V., Y.T. Maeda, and A. Libchaber. 2011. Development of an artificial cell, from self-organization to computation and self-reproduction. *Proceedings of the National Academy of Science U S A*, 108:3473-3480.

3.Luisi, P.L., F. Ferri, and P. Stano. 2006. Approaches to semi-synthetic minimal cells: a review. *Naturwissenschaften,* 93:1-13.

4.Jewett, M.C. and A.C. Forster. 2010. Update on designing and building minimal cells. *Current opinion in biotechnology,* 21:697-703.

5.Shin, J. and V. Noireaux. 2012. An E. coli cell-free expression toolbox: application to synthetic gene circuits and artificial cells. *ACS synthetic biology,* 1:29-41.

6.Caschera, F., T. Sunami, T. Matsuura, H. Suzuki, M.M. Hanczyc, and T. Yomo. 2011. Programmed vesicle fusion triggers gene expression. *Langmuir : the ACS journal of surfaces and colloids,* 27:13082-13090.

7.Caschera, F., S. Rasmussen, and M.M. Hanczyc. 2013. An Oil Droplet Division-Fusion Cycle. *ChemPlusChem, 78*:52-54.

8.Caschera, F., P. Stano, and P.L. Luisi. 2010. Reactivity and fusion between cationic vesicles and fatty acid anionic vesicles. *Journal of colloid and interface science,* 345:561-565.

9.Jewett, M.C., B.R. Fritz, L.E. Timmerman, and G.M. Church. 2013. In vitro integration of ribosomal RNA synthesis, ribosome assembly, and translation. *Molecular systems biology,* 9:678.

10.Fritz, B.R. and M.C. Jewett. 2014. The impact of transcriptional tuning on in vitro integrated rRNA transcription and ribosome construction. *Nucleic acids research,* 42:6774-6785.

11.Liu, Y., B.R. Fritz, M.J. Anderson, J.A. Schoborg, and M.C. Jewett. 2014. Characterizing and Alleviating Substrate Limitations for Improved in vitro Ribosome Construction. *ACS synthetic biology*.

12.Szostak, J.W., D.P. Bartel, and P.L. Luisi. 2001. Synthesizing life. *Nature* 409:387-390.

13.Hanczyc, M.M., S.M. Fujikawa, and J.W. Szostak. 2003. Experimental models of primitive cellular compartments: encapsulation, growth, and division. *Science,* 302:618-622.

14.Caschera, F., M.A. Bedau, A. Buchanan, J. Cawse, D. de Lucrezia, G. Gazzola, M.M. Hanczyc, and N.H. Packard. 2011. Coping with complexity: machine learning optimization of cell-free protein synthesis. *Biotechnology and bioengineering,* 108:2218-2228.

15.Caschera, F., G. Gazzola, M.A. Bedau, C. Bosch Moreno, A. Buchanan, J. Cawse, N. Packard, and M.M. Hanczyc. 2010. Automated discovery of novel drug formulations using predictive iterated high throughput experimentation. *PloS one, 5*:e8546.

18.Caschera, F. and V. Noireaux. 2014. Synthesis of 2.3 mg/ml of protein with an all Escherichia coli cell-free transcription-translation system. *Biochimie,* 99C:162-168.

19.Caschera, F. and V. Noireaux. 2014. A Cost-Effective Polyphosphate-Based Metabolism Fuels an All E. coli Cell-Free Expression System, *Metabolic Engineering*, 27:187-189.

20.Caschera, F. and V. Noireaux. 2015. Preparation of amino acid mixtures for cell-free expression systems. *Biotechniques*, 58:40-43.

21.Uno, K., T. Sunami, N. Ichihashi, Y. Kazuta, T. Matsuura, and T. Yomo. 2014. The evolutionary enhancement of genotype-phenotype linkages in the presence of multiple copies of genetic material. *Chembiochem,* 15:2281-2288.

22.Ichihashi, N., T. Matsuura, H. Kita, T. Sunami, H. Suzuki, and T. Yomo. 2010. Constructing partial models of cells. *Cold Spring Harbor Perspectives in Biology,* 2:a004945.

Proceedings Paper

16.Caschera, F., M.M. Hanczyc, and S. Rasmussen. 2011. Machine learning for drug design, molecular machines and evolvable artificial cells. *GECCO '11 Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, 831-832.

17.Caschera, F. and V. Noireaux. 2014. A novel in vitro metabolic scheme for the construction of a minimal biological cell. *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems, 14*:3-5.

The Universality of Peer-Influence in Social Networks

Flávio L. Pinheiro[1,2,3,4], Marta D. Santos[5], Francisco C. Santos[1,4] and Jorge M. Pacheco[2,6,4]

[1] INESC-ID & Instituto Superior Técnico, Universidade de Lisboa, 2744-016 Porto Salvo, Portugal
[2] Centro de Biologia Molecular e Ambiental da Universidade do Minho, 4710-057 Braga, Portugal
[3] Centro de Física da Universidade do Minho, 4710-057 Braga, Portugal
[4] ATP-group, CMAF, Instituto para a Investigação Interdisciplinar, P-1649-003 Lisboa, Portugal
[5] Departamento de Física & I3N, Universidade de Aveiro, 3810-193 Aveiro, Portugal
[6] Departamento de Matemática e Aplicações da Universidade do Minho, 4710-057 Braga, Portugal
flavio.lpp@gmail.com

Social networks pervade our everyday lives: we interact, influence and are influenced by our friends and acquaintances. The recent availability of large amounts of data on social networks has fostered quantitative analyses of the distribution of information on them, including behavioural traits and fads. In particular, recent studies have shown the existence of positive correlations in the distribution of traits in a social network composed by the participants of the *Framingham Heart* study Christakis and Fowler (2007); Fowler and Christakis (2008). Surprisingly the peer-influence patterns found among the participants went beyond the influence of their closest peers, but also their friends' friends,up to three degrees of influence.

In Pinheiro et al. (2014) we show how similar patterns of correlations between peers emerge in networked populations through standard models (yet reflecting intrinsically different mechanisms) of information spreading such as the Voter's Model, the SIR epidemic model (see Fig. 1) and Evolutionary Game Theory models of cooperation. We argue that empirically observed patterns of correlation among peers emerge naturally from a wide range of dynamical processes, being essentially independent of the type of information, on how it spreads, and even on the class of underlying network that inter-connects individuals. Finally, we show that the sparser and clustered the network, the more far-reaching the influence of each individual will be.

Acknowledgments

Financial support from FCT-Portugal is gratefully acknowledged, through scholarship SFRH/BD/77389/2011; grants PTDC/MAT/122897/2010, EXPL/EEI-SII/2556/2013 and by multi-annual funding of CMAF-UL, CBMA-UM and INESC-ID (under the projects PEst-C/BIA/UI4050/2011 and UID/CEC/50021/2013) also provided by FCT-Portugal.

References

Christakis, N. A. and Fowler, J. H. (2007). The spread of obesity in a large social network over 32 years. *New England Journal of Medicine*, 357(4):370–379.

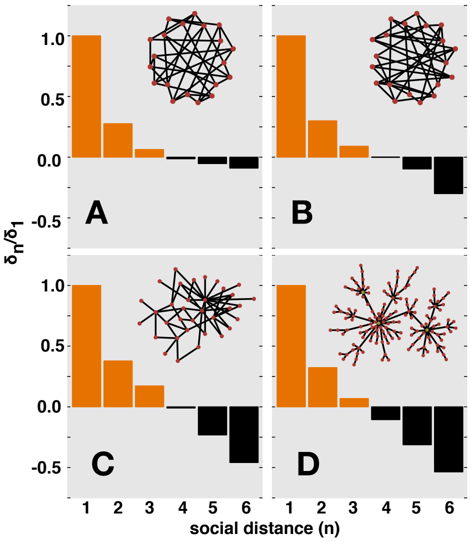


Figure 1: Peer-influence patterns obtained from a *Susceptible-Infected-Recovered* epidemics model on four different types of population structure: (**A**) *Homogeneous Small-World*; (**B**)*Heterogeneous Small-World*; (**C**) *Exponential* and (**D**) *Scale-Free* networks. Results correspond to the normalised correlations (δ_n/δ_1) with respect to values obtained for the closest neighbours (δ_1). Correlations (δ_n) measure how probable it is to find an individual with the same trait as a focal node at a social distance of n links. Orange/Black bars denote positive/negative correlations measured at a social distance of n and relative to the expected values in a random distribution of traits.

Fowler, J. H. and Christakis, N. A. (2008). Dynamic spread of happiness in a large social network: longitudinal analysis over 20 years in the framingham heart study. *BMJ: British Medical Journal*, 337.

Pinheiro, F. L., Santos, M. D., Santos, F. C., and Pacheco, J. M. (2014). Origin of peer influence in social networks. *Physical Review Letters*, 112(9):098702.

Programs as Polypeptides

Lance R. Williams[1]

[1]Department of Computer Science, University of New Mexico, Albuquerque, NM 87131
williams@cs.unm.edu

Abstract

We describe a visual programming language for defining behaviors manifested by reified actors in a 2D virtual world that can be compiled into programs comprised of sequences of combinators that are themselves reified as actors. This makes it possible to build programs that build programs from components of a few fixed types delivered by diffusion using processes that resemble chemistry as much as computation.

Introduction

Self-replicating programs have been defined using computational models that vary in *expressiveness* and *verisimilitude*. If we adopt the definition used in the field of programming languages (Felleisen, 1990), then expressiveness varies along a spectrum that begins with *cellular automata* (CA) defined using lookup tables, increases with *artificial chemistries* based on symbol rewrite rules, and peaks in (more or less) conventional programming languages (which themselves vary along a spectrum that begins with machine language and ends in high-level languages like Lisp).

By verisimilitude, we mean providing an interface with the affordances and limitations of a natural physics. Models with high verisimilitude define *virtual worlds*. Because CAs are spatially embedded and governed by simple rules defined on local neighborhoods, most would say that the verisimilitude of CAs is high. However, since state is updated everywhere synchronously, and this (unlike a natural physics) requires a global clock, CAs are not *indefinitely scalable* (Ackley, 2013). Because *asynchronous cellular automata* (ACA) do not suffer from this limitation yet are just as powerful (Nakamura, 1974; Berman and Simon, 1988; Nehaniv, 2004), ACAs are the gold standard in virtual worlds.

Many artificial chemistries lack verisimilitude because the symbols that the rewrite rules transform are not embedded in any physical space (Berry and Boudol, 1990; Paun, 1998; Fontana and Buss, 1999). Others have far greater resemblance to real physical systems (Laing, 1977; Smith et al., 2003; Hutton, 2004). These assign symbols to positions in a virtual world, restrict interactions to local neighborhoods, and rely on diffusion for data transport.

Programs written in conventional programming languages generally require a *random access stored program* (RASP) computer to host them.[1] Because of *program-data equivalence*, RASPs permit relatively simple solutions to the self-replication problem based on *reflection.* Yet self-replicating programs written in conventional programming languages are (in effect) stuck in boxes; it makes no difference whether it is one big box (Ray, 1994) or many little boxes interacting in a virtual world (Adami et al., 1994); because they read, write, and reside in random access memories, the programs themselves are fundamentally non-physical.

In the game of defining virtual worlds and creating self-replicating programs inside those worlds, there is a tradeoff between the non-contingent complexity of *physical law* and the purely contingent complexity of the *initial conditions* that define a program and its self-description. We propose that the ratio of contingent and non-contingent complexity is positively correlated with the property that Pattee (1995) calls *semantic closure*. Ideally, we would like to pursue an approach that combines the expressiveness of conventional programming languages with the physical verisimilitude of ACAs while maximizing the ratio of contingent and non-contingent complexity. To do this, we need to break programs out of their boxes; we need reified programs that assemble copies of themselves from reified building blocks; we need to imagine *programs as polypeptides.*

Superficially, there is a similarity between the sequences of instructions that comprise a machine language program and the sequences of nucleotides and amino acids that comprise the biologically important family of molecules known as *biopolymers*. It is tempting to view all of these sequences as ‘programs,’ broadly construed. However, machine language programs and biopolymers differ in (at least) one significant way, and that is the number of elementary building blocks from which they are constructed. The nucleotides that comprise DNA and RNA are only of four types; the amino acids that comprise polypeptides are only of twenty; and while bits might conceivably play the passive represen-

[1] See Williams (2014) for a notable exception.

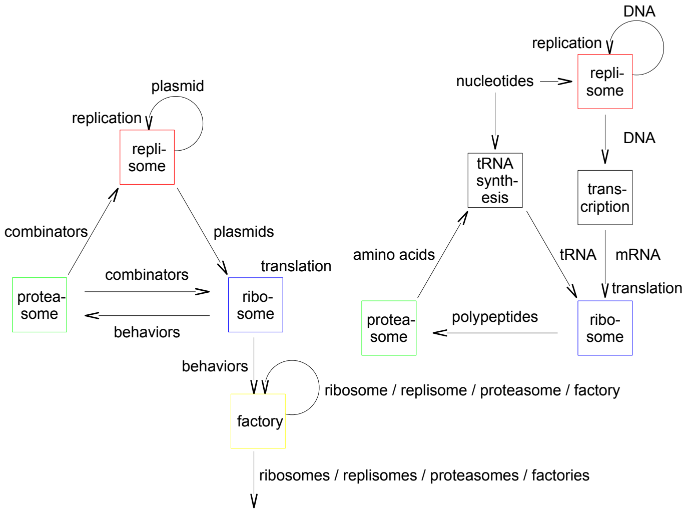


Figure 1: Framework proposed in this paper (left). Fundamental dogma of molecular biology (right).

tational role of nucleotides, they can not play the active functional role of amino acids; this role can only be played by instructions. While the instruction set of a simple RASP can be quite small, the number of distinct *operands* that (in effect) modify the instructions is a function of the word size of the machine, and is therefore (at a minimum) in the thousands.[2] The implication for the study of self-replicating programs is profound: while biopolymers can be assembled by physical processes from building blocks of a few fixed types, it is impossible to construct machine language programs for a RASP this way.

DNA and RNA are copiable, transcribable and translatable descriptions of polypeptides. DNA is (for the most part) chemically inert while polypeptides are chemically active. Polypeptides can not serve as representations of themselves (or for that matter of anything at all) because their enzymatic functions render this impossible. Information flows in one direction only. Watson and Crick (1953) thought this idea so important that they called it "the fundamental dogma of molecular biology." It is the antithesis of the program-data equivalence which makes reflection possible. See Figure 1.

Combinators are functions with no free variables. In this paper we show how programs in a visual programming language just as expressive as machine language can be compiled into sequences of combinators of only forty two types. Where machine language programs would use iteration, the programs that we compile into combinators employ *nondeterminism*. The paper culminates in the experimental demonstration of a *computational ribosome*, a 'machine' in a 2D virtual world that assembles programs from combinators using inert descriptions of programs (also comprised of combinators) as templates.

Reified Actors

Actors are created using three different constructors: $[\,]^-$ creates *combinators*, $[\,]^+$ creates *behaviors*, and $[\,]_k$ creates *objects*. Like amino acids, which can be composed to form polypeptides, *primitive* combinators can be composed to form *composite* combinators. Behaviors are just combinators that have been repackaged with the $[\,]^+$ constructor. Prior to repackaging, combinators do not manifest their function; this might correspond (in our analogy) to the folding of a polypeptide chain into a *protein*.

Objects are containers that can contain other actors. Each is one of four immutable types: $[\,]_0$, $[\,]_1$, $[\,]_2$ and $[\,]_3$. For example, $[x, y, z]_2$ is an object of type two that contains three actors, x, y and z. Primitive combinators and empty objects have unit *mass*. The mass of a composite combinator is the sum of the masses of the combinators of which it is composed. The mass of an object is the sum of its own mass and the masses of the actors it contains. Since actors can neither be created nor destroyed, mass is conserved.

Actors are reified by assigning them positions in a 2D virtual world. Computations progress when actors interact with other actors in their 8-neighborhoods by means of the behaviors they manifest. All actors are subject to *diffusion*. An actor's diffusion constant decreases inversely with its mass. This reflects the real cost of data transport in the (notional) ACA substrate. Multiple actors can reside at a single site, but diffusion never moves an actor to an adjacent occupied site if there is an adjacent empty site.

As with *membranes* in Paun (1998), objects can be nested to any level of depth. The object that contains an actor (with no intervening objects) is termed the actor's *parent*. An actor with no parent is a *root*. Root actors (or actors with the same parent) can be associated with one another by means of *groups* and *bonds*. Association is useful because it allows working sets of actors to be constructed and the elements of these working sets to be addressed in different ways.

The first way in which actors can associate is as members of a *group*. All actors belong to exactly one group and this group can contain a single actor. For this reason, groups define an *equivalence relation* on the set of actors. A group of root actors is said to be *embedded*. All of the actors in an embedded group diffuse as a unit and all behaviors manifested by actors in an embedded group (or contained inside such an actor) share a finite time resource in a zero sum fashion. Complex computations formulated in terms of large numbers of actors manifesting behaviors inside a single object or group will therefore be correspondingly slow. Furthermore, because of its large net mass, the object or group that contains them will also be correspondingly immobile.

The second way in which actors can associate is by *bonding*. Bonds are short relative addresses that are automatically updated as the actors they link undergo diffusion. Because bonds are short (L_1 distance less than or equal to two), they restrict the diffusion of the actors that possess

[2] Although they play many roles in machine language programs, non-register operands are generally *addresses*.

them. Undirected bonds are defined by the *hand* relation H, which is a *symmetric relation* on the set of actors, *i.e.,* $H(x,y) = H(y,x)$. Directed bonds are defined by the *previous* and *next* relations, P and N, which are *inverse relations* on the set of actors, *i.e.,* $P(x,y) = N(y,x)$.

If the types of combinators and behaviors were defined by the sequences of primitive combinators of which they are composed, then determining type equivalence would be relatively expensive. For this reason, we chose instead to define type using a simple recursive hash function that assigns combinators with distinct multisets of components to distinct types: the hash values of composite combinators are defined as the product of the hash values of their components; primitive combinators have hash values equal to prime numbers.[3] Type equivalence for behaviors is defined in the same way, the types of combinators and behaviors being distinct due to the use of different constructors. Although this hash function is (clearly) not collision free, it is quite good and it has an extremely useful property, namely, that composite combinators can be broken down (literally decomposed) into their primitive components by prime factorization.[4]

Apart from composition, containment, group and bonds there is no other mutable persistent state associated with actors. In particular, there are no integer registers. Primitive combinators exist for addressing individual actors or sets of actors using most of these relations. These, and other primitive combinators for modifying actors' persistent states will be described later.

Non-deterministic Comprehensions

Sets can be converted into *superpositions* using the non-deterministic choice operator (McCarthy, 1963):

$$\begin{aligned} \text{amb}\,\{\} &= \langle\,\rangle \\ \text{amb}\,\{x, y \ldots\} &= \langle x, y \ldots\rangle. \end{aligned}$$

When *amb* is applied to a non-empty set, it causes the branch of the non-deterministic computation that called *amb* to fork. Conversely, empty sets cause the branch to fail. When a branch fails, the deterministic implementation backtracks.

Monads are an abstract datatype that allows programmers to define rules for composing functions that deviate from mathematically pure functions in prescribed ways. Multivaluedness (represented by sets) and non-determinism (represented by superpositions) are just two examples. The monad interface is defined by two operations called *unit* and *bind*. *Unit* transforms ordinary values a into *monadic values*, *e.g.,* $\text{unit}_A\, x = \langle x\rangle$ where A is the superposition monad. Functions like *unit* that take ordinary values and return monadic values are termed *monadic functions. Bind* (the infix operator '$\gg\!=$' in Haskell) allows monadic functions to be applied to monadic values. This permits monadic functions to be chained; the output of one provides the input to the next.

Monads are intimately related to set builder notation or *comprehensions*. By way of illustration, consider the following non-deterministic comprehension that fails if n is prime and returns a (non-specified) factor of n if n is composite:

$$\langle\, x \mid x \in \langle 1\,..\,n-1\rangle\,,\; y \in \langle 1\,..\,x\rangle\,,\; x\,y \;=\; n\,\rangle\,.$$

Wadler (1990) showed that notation like the above is syntactic sugar for monadic expressions and described a process for translating the former into the latter. Comprehension *guards*, *e.g.,* $x\,y = n$, are translated using the function

$$\begin{aligned} \text{guard}_M\ \mathit{True} &= \text{unit}_M\ \bot \\ \text{guard}_M\ \mathit{False} &= \text{zero}_M \end{aligned}$$

where M is the monad and $\bot$ is *undefined.* Because zero_A is $\langle\,\rangle$, if guard_A is applied to *False*, the branch of the computation that called guard_A fails. Conversely, if guard_A is applied to *True*, the branch continues. Using this device, the primality comprehension can be desugared as follows

$$\lambda n \rightarrow (\text{unit}_A\, n \overset{A}{\gg\!=} \text{unit}_A \cdot (-\,1) \overset{A}{\gg\!=} \text{amb} \cdot \iota \overset{A}{\gg\!=}$$

$$\lambda x \rightarrow (\text{unit}_A\, x \overset{A}{\gg\!=} \text{amb} \cdot \iota \overset{A}{\gg\!=} \text{unit}_A \cdot (\times\, x) \overset{A}{\gg\!=}$$

$$\text{unit}_A \cdot (=\, n) \overset{A}{\gg\!=} \text{guard}_A \overset{A}{\gg\!=} \text{unit}_A\, x))$$

where $(\iota\, x)$ equals $\{1\,..\,x\}$.

From Comprehensions to Dataflow Graphs

Recall that our goal is to create programs comprised solely of combinators. To maximize composability, these combinators should be of a single type, yet the desugared comprehension above contains functions of many different types. However, if sets are used to represent sets, singleton sets are used to represent scalars, and non-empty and empty sets are used to represent *True* and *False*, then the type signatures

$$\begin{aligned} \rightarrow\boxed{f'}\rightarrow \quad &:: \quad \{a\} \rightarrow \langle\{a\}\rangle \\ \substack{\rightarrow\\ \rightarrow}\boxed{g'}\rightarrow \quad &:: \quad \{a\} \rightarrow \{a\} \rightarrow \langle\{a\}\rangle \end{aligned}$$

are general enough to represent the types of all functions in the desugared comprehension. To prove this, we first show that *amb* can be lifted to the type, $\{a\} \rightarrow \langle\{a\}\rangle$, as follows:

$$\begin{aligned} \text{amb}'\,\{\} &= \langle\,\rangle \\ \text{amb}'\,\{x, y\ldots\} &= \langle\{x\},\{y\}\ldots\rangle. \end{aligned}$$

We then devise a way to lift functions like ι with type, $a \rightarrow \{a\}$. This is accomplished using the bind operator $(\gg\!=_S)$ for the set monad S. The bind operator behaves like this

$$\{\,x, y \ldots\,\} \overset{S}{\gg\!=} f = f\,x \cup f\,y \cup \ldots$$

[3] We could instead use nested objects to label combinators so that they can be compared. This would be like using codons constructed from nucleotides to label amino acids in transfer RNAs.

[4] This is analogous to the function in the cell which is performed by the molecular assemblies called proteasomes and in the organelles called lysosomes.

and can be defined as follows

$$(\overset{S}{\gg=}\ f) = \text{join}_S \cdot\ (\text{map}_S\, f)$$

where join_S is right fold of $(\cup)$ and

$$\text{map}_S\ f\ \{x, y\ldots\} = \{f\,x,\ f\,y\ \ldots\}.$$

Bind can then be used with unit_A to lift ι into a function

$$\iota' = \text{unit}_A \cdot (\overset{S}{\gg=}\ \iota)$$

with the type, $\{a\} \rightarrow \langle\{a\}\rangle$, as demonstrated below

$$\iota'\{x, y\ldots\} \quad = \quad \langle\, \iota\, x \cup \iota\, y \cup \ldots\rangle.$$

Next we define two functions of type, $\{a\} \rightarrow \langle\{a\}\rangle$, to replace *guard*. The first causes a computation to fail when its argument is empty while the second does the opposite:

$$\begin{aligned} \text{some}'\,\{\} \quad &= \quad \langle\,\rangle \\ \text{some}'\,\{x, y\ldots\} \quad &= \quad \langle\{x, y\ldots\}\rangle \\ \text{none}'\,\{\} \quad &= \quad \langle\{\}\rangle \\ \text{none}'\,\{x, y\ldots\} \quad &= \quad \langle\,\rangle. \end{aligned}$$

Finally, the desugared comprehension contains functions like (-1), $(\times)$ and $(=)$ that map scalars to scalars, yet we need functions that map sets to superpositions of sets. Fortunately, sensible lifted forms for these functions are easily defined. For example

$$\begin{aligned} \text{pred}' \quad &= \quad \text{unit}_A \cdot (\text{map}_S\,(-1)) \\ \text{times}'\,x'\,y' \quad &= \quad \text{unit}_A\ \{\, x \times y \mid x \in x',\ y \in y'\,\} \\ \text{equals}'\,x'\,y' \quad &= \quad \text{unit}_A\ \{\, x \mid x \in x',\ y \in y',\ x = y\,\} \end{aligned}$$

where x' and y' are of type $\{a\}$. Using these lifted functions and those defined previously, the non-deterministic comprehension for deciding primality can be translated as follows:

$$\lambda n' \rightarrow (\text{unit}_A\, n'\ \overset{A}{\gg=}\ \text{pred}'\ \overset{A}{\gg=}\ \iota'\ \overset{A}{\gg=}\ \text{amb}'\ \overset{A}{\gg=}$$

$$\lambda x' \rightarrow (\text{unit}_A\, x'\ \overset{A}{\gg=}\ \iota'\ \overset{A}{\gg=}\ \text{amb}'\ \overset{A}{\gg=}$$

$$(\text{times}'\ x')\ \overset{A}{\gg=}\ (\text{equals}'\ n')\ \overset{A}{\gg=}\ \text{some}'))$$

where n' is of type $\{a\}$. This was a lot of work, but we have reaped a tangible benefit, namely, non-deterministic comprehensions can now be rendered as *dataflow graphs*. In Figure 2 (top) boxes with one input have type signatures matching f' and boxes with two inputs have type signatures matching g'. Arrows connecting pairs of boxes are instances of $(\gg=_A)$. Junctions correspond to values of common subexpressions bound to variable names introduced by λ–expressions. Lastly, Ⓐ is *amb*$'$ and Ⓢ is *some*$'$. This result is important because, without the amenity (provided by all general purpose programming languages) of being able to define and name functions, comprehension syntax quickly becomes unwieldy. For this reason, we make extensive use of dataflow graphs as a visual programming language in the remainder of this paper.

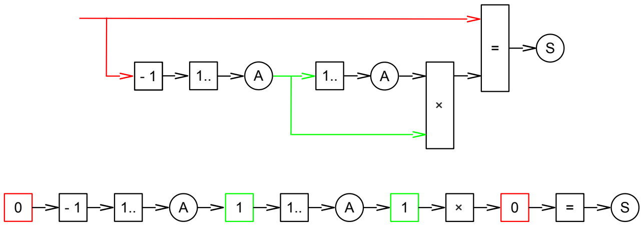


Figure 2: *Non-deterministic dataflow graph* for deciding primality (top). Dataflow graph compiled into a sequence of *non-deterministic combinators* (bottom).

From Dataflow Graphs to Combinators

One might assume that evaluation of dataflow graphs containing junctions would require an interpreter with the ability to create and apply anonymous functions or *closures*. These would contain the environments needed to lookup the values bound to variable names introduced by λ–expressions. Happily, this turns out to be unnecessary. In this section we show how dataflow graphs can be evaluated by a stack machine and define a set of combinators that can be used to construct stack machine programs.

In general, combinators apply functions to one (or two) values of type $\{a\}$ popped from the front of the stack and then push a result of type $\{a\}$ back onto the stack. Since dataflow graphs are non-deterministic, the stack machine is also. This means that each combinator f'' transforms a stack of sets into a superposition of stacks of sets

$$\rightarrow\boxed{f''}\rightarrow\ \ ::\ \ [\{a\}] \rightarrow \langle[\{a\}]\rangle.$$

Unary operators f' can be converted to combinators of type f'' as follows:

$$f''\ \ (x' : s'') = \text{map}_A\ (:\ s'')\ (f'\,x')$$

where stack s'' is of type $[\{a\}]$, map_A maps functions over superpositions and $(:\ s'')$ is the function that pushes sets onto the front of s''. Note that f'' does not change the length of the stack; it consumes one value and leaves one value behind. Binary operators g' can also be converted to combinators of type f'' as follows:

$$g''\ \ (x' : y' : s'') = \text{map}_A\ (:\ s'')\ (g'\,x'\,y').$$

Note that g'' decreases the length of the stack by one; it consumes two values and leaves one value behind. The combinator forms of *some*$'$ and *none*$'$ are slightly different; they do not push a result onto the stack. Instead, they pop the stack when a non-deterministic computation has yielded a satisfactory intermediate result (whether that is something or nothing) and fail otherwise:

$$\text{some}''\ (x' : s'') = \begin{cases} \langle\,\rangle & \text{if } x' = \{\} \\ \text{unit}_A\ s'' & \text{otherwise} \end{cases}$$

$$\text{none}''\,(x' : s'') = \begin{cases} \text{unit}_A\; s'' & \text{if } x' = \{\} \\ \langle\,\rangle & \text{otherwise.} \end{cases}$$

Multiple functions can be applied to a single value by pushing copies of the value onto the top of the stack and then applying the functions to the copies. This preserves the value for future use and eliminates the need for closures. Accordingly, we define a set of combinators that copy and push values located at different positions within the stack

$$\text{x}''_k\,(s'') \;= \text{unit}_A\,((s''\;!!\;(n-k)) : s'')$$

where $k \in \{0..9\}$, (!!) returns the element of a list with a given index, and n is the length of s''. With this last puzzle piece in place, we can finally do what we set out to do, namely, compile the comprehension for deciding primality into a sequence of combinators

$$\text{x}''_0 \overset{A}{>\!=\!>} \text{pred}'' \overset{A}{>\!=\!>} \iota'' \overset{A}{>\!=\!>} \text{amb}'' \overset{A}{>\!=\!>} \text{x}''_1 \overset{A}{>\!=\!>} \iota'' \overset{A}{>\!=\!>} \text{amb}''$$

$$\overset{A}{>\!=\!>} \text{x}''_1 \overset{A}{>\!=\!>} \text{times}'' \overset{A}{>\!=\!>} \text{x}''_0 \overset{A}{>\!=\!>} \text{equals}'' \overset{A}{>\!=\!>} \text{some}''$$

where $(>\!=\!>)$ is *Kleisli composition*

$$f >\!=\!> g = (>\!>= g) \cdot f$$

In Figure 2 (bottom) boxes are functions with type signatures matching f''. Arrows connecting pairs of boxes are instances of $(>\!=\!>_A)$. Lastly, Ⓐ is *amb″* and Ⓢ is *some″*.

Reified Actor Comprehensions

The last two sections of the paper demonstrated that: 1) Non-deterministic comprehensions can be represented as dataflow graphs; and 2) Dataflow graphs can be compiled into sequences of combinators that evaluate comprehensions by transforming the state of an abstract machine. In this section we describe a visual programming language for specifying behaviors manifested by reified actors in a virtual world. All results from prior sections apply. However, non-determinism must be combined with other effects to construct a monad more general than A which we call R (for *reified actor*). In addition to representing superpositions, monad R provides mutation of a threaded global state and data logging so that behaviors composed of combinators can report the time they consume. The boxes of dataflow graphs with one and two inputs now have types $\{Actor\} \rightarrow \langle\{Actor\}\rangle'$ and $\{Actor\} \rightarrow \{Actor\} \rightarrow \langle\{Actor\}\rangle'$ where $\langle\,\rangle'$ is the type constructor of monad R. Arrows connecting boxes are instances of $(>\!>=_R)$. Combinators now have type $[\{Actor\}] \rightarrow \langle[\{Actor\}]\rangle'$ and are composed with $(>\!=\!>_R)$.

Combinators can be divided into the categories: *generators*, *guards*, *relations*, and *actions*. Generators are unary operators that characterize sets of actors using the devices of groups, containment, bonds, and neighborhood (Table 1). They can be composed to address different sets. For example, an actor's siblings can all be addressed using the subgraph $\boxed{\hat{}} \rightarrow \boxed{@}$. Generators can also be composed with guards (Table 2). This can be used either to address single actors or to specify preconditions for actions. For example, the subgraph $\boxed{\hat{}} \rightarrow \boxed{@} \rightarrow$ Ⓐ addresses a single sibling while the subgraph $\boxed{\#} \rightarrow$ Ⓝ fails if the actor has a neighbor.

Table 1: Unary generators.

Name	Abbrev.	Definition
hands	|	actor sharing hand with x
nexts	>	actor with directed bond from x
prevs	<	actor with directed bond to x
bonds	:	union of hands, nexts and prevs
neighbors	#	actors in neighborhood of x
contents	@	actors that are contained in x
parents	ˆ	actor that contains x
members	*	members of group of x
others	+	members of group of x but not x

Table 2: Unary guards.

Name	Abbrev.	Definition
amb	A	non-deterministic choice
some	S	Fail if empty.
none	N	Fail if non-empty.

Relations exist for testing equality and type equivalence (Table 3). They are binary operators and are generally applied to singleton sets in combination with guards to specify preconditions for actions. When applied to non-singleton sets, the equality operator and its negation compute set intersection and difference.

Table 3: Binary relations.

Name	Abbrev.	Definition
same	=	set intersection
different	! =	set difference
similar	∼	all x type equivalent to some y
dissimilar	! ∼	all x type equivalent to no y

Actions for modifying actors' persistent states are the final category of boxes in dataflow graphs. Actions are rendered as grey boxes and are executed only after all non-actions have been evaluated and only if no guard has failed. All actions are reversible but the masses and types of primitive combinators and empty objects are immutable. The full set of unary and binary actions is shown in Tables 4 and 5.

Where data dependencies determine order of execution, this order is followed. Where it would otherwise be underdetermined, two devices are introduced to specify execution

Table 4: Unary actions.

Name	Abbrev.	Definition
drop	! |	Delete hand of x.
unbond	! >	Delete directed bond from x.
unbond′	! <	Delete directed bond to x.
quit	∗ −>	Remove x from its group.
exit	@ −>	Place x inside its parent's parent.
digest	>!>	Reduce x to primitive combinators.
on	/	Replace combinator with behavior.
off	\	Replace behavior with combinator.

Table 5: Binary actions.

Name	Abbrev.	Definition
grab	|	Create hand between x and y.
bond	>	Create directed bond from x to y.
bond′	<	Create directed bond from y to x.
join	−> ∗	x joins group of y.
eat	−>@	Place x inside y.
compose	>=>	Replace x with x $>\!=\!>_R$ y.
swap	%	x and y swap positions and bonds.

order. First, all actions return their first (or only) argument if they succeed. This allows one action to provide the input to a second and (when employed) introduces a data dependency that determines execution order. Second, execution order can be explicitly specified using dotted *control lines*.

In addition to non-determinism and mutable threaded state, instances of monad R also possess a data logging ability that is used to instrument combinators so that behaviors comprised of them can report the time they consume. Because the unit of time is one primitive operation of the abstract machine, most primitive combinators increase logged time by one when they are run. Significantly, this occurs on all branches of the non-deterministic computation until a branch succeeds so that the full cost of simulating non-determinism on a (presumed) deterministic substrate by means of backtracking is accounted for. Two kinds of combinators increase logged time by amounts other than one. Since the time required to compute set intersections and differences is the product of the sets' lengths, for binary relations, the logged time is increased by this value instead (which equals one in the most common case of singleton sets). Finally, actions that change the position of an actor, *e.g.*, *join*, pay an additional time penalty proportional to the product of the actor's mass and the L_1 distance moved.

Ideally, the actor model described in this paper would be reified as an ACA so that self-replicating programs consume real physical resources. Actors in an embedded group might share a single processor or might jointly occupy a 2D area of fixed size that collects a fixed amount of light energy per unit time. The effect would be the same; the number of primitive abstract machine operations executed per unit time by the processor (or in the area) would be fixed.

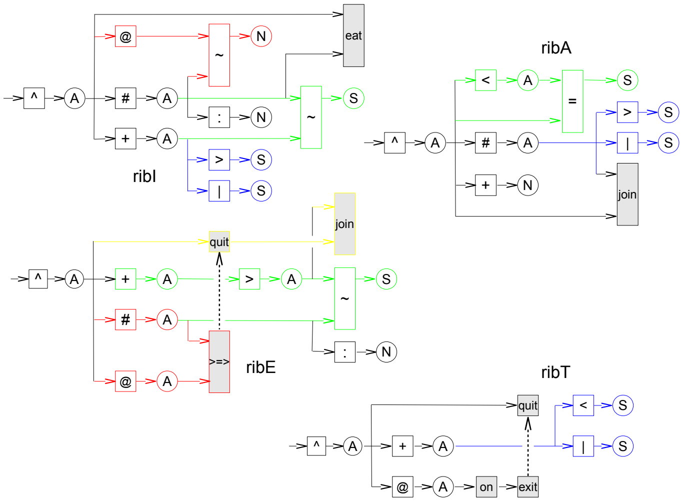


Figure 3: Behaviors defining a ribosome.

For the time being, we implement the reified actor model as an event driven simulation using a priority queue (Gillespie, 1977). Event times are modeled as Poisson processes associated with embedded groups and event rates are consistent with the joint consumption by actors in groups of finite time resources. Events are of two types. When a *diffusion event* is at the front of the queue, the position of the group in its neighborhood is randomly changed (as previously described). Afterwards, a new diffusion event associated with the same group is enqueued. The time of the new event is a sample from a distribution with density $f_D(t) = D\,e^{-Dt/(m\,s)}/(m\,s)$ where m is mass, s is distance, and D is the ratio of the time needed to execute one primitive operation and the time needed to transport a unit mass a unit distance. As such, it defines the relative cost of computation and data transport in the ACA substrate.[5]

When an *action event* is at the front of the queue, a behavior is chosen at random from among all actors of type behavior in the group. After the behavior is executed, the time assigned to the new action event is a sample from a distribution with density $f_A(t) = e^{-t/c}/c$ where c is the time consumed by the behavior.

Computational Ribosomes

Biological enzymes can be reified as chains of nucleotides or amino acids. The first can be read and copied but are spatially distributed and purely representational; the second are representationally opaque but compact and metabolically active. Comprehensions can be compiled into sequences of primitive combinators and reified in analogous ways. A

[5] In all of our experiments D equals 10.

plasmid is a compiled comprehension reified as a chain of actors of type combinator linked with directed bonds:

$$P = \llbracket c_0 \rrbracket^- > \llbracket c_1 \rrbracket^- > \cdots > \llbracket c_{N-1} \rrbracket^-$$

where $(>)$ is a directed bond and $\llbracket\ \rrbracket$ denotes an actor that is reified at the root level. A single undirected bond (not shown) closes the chain and marks the plasmid's *origin*. While plasmids are spatially distributed chains of many actors, *enzymes* are single actors of type behavior:

$$E = \llbracket c_0 \ >\!\!\!\Longrightarrow_R \ c_1 \ >\!\!\!\Longrightarrow_R \cdots \ >\!\!\!\Longrightarrow_R c_{N-1} \rrbracket^+.$$

Biological ribosomes are arguably the most important component of the fundamental dogma (Watson and Crick, 1953). They translate messenger RNA into polypeptides using a four stage process of *association*, *initiation*, *elongation* and *termination*. We can construct a *computational ribosome* that will translate plasmids into enzymes by defining four behaviors with analogous functions (Figure 3), reifying the behaviors as enzymes, and placing them inside an actor of type object

$$R = \llbracket E_{\mathrm{ribA}}, E_{\mathrm{ribI}}, E_{\mathrm{ribE}}, E_{\mathrm{ribT}} \rrbracket_0.$$

Behavior *ribA* first checks to see if R possesses a self-directed bond.[6] If so, *ribA* attaches R to the plasmid by adding it to the group of the initial combinator, $\llbracket c_0 \rrbracket^-$. Next, *ribI* finds an actor in the neighborhood with type matching $\llbracket c_0 \rrbracket^-$ and places it inside R. When R is at position n on the plasmid, *ribE* finds a neighbor with type matching $\llbracket c_{n+1} \rrbracket^-$ and composes it with the combinator inside R, *i.e.,* with $[c_0 \ >\!\!\!\Longrightarrow_R \cdots \ >\!\!\!\Longrightarrow_R c_n]^-$. It then advances the position of R to $n+1$. This process continues until R reaches $N-1$, at which point *ribT* promotes the combinator to a behavior, expels it, and detaches R from the plasmid.

If a ribosome and a plasmid are placed in the world with a supply of primitive combinators, the ribosome manufactures the enzyme described by the plasmid

$$R + P_b + \textstyle\sum_C m_b(c) \, \llbracket c \rrbracket^- \rightarrow R + P_b + E_b$$

where C is the set of 42 primitive combinators and $m_b(c)$ is the number of combinators of type c in P_b and E_b, *i.e.,* the plasmid and enzyme reifications of behavior b.

Now that we have a ribosome, we need something to do with it. We could (of course) use ribosomes to synthesize the enzymes of which they themselves are comprised. However, it would be more interesting if these enzymes were then used to construct additional ribosomes. To accomplish this, we need a 'machine' that will collect the finished enzymes and place them inside an object of the correct type. We call this machine a *factory*. Factories are copiers of *compositional information*, which is heritable information distinct from the *genetic information* that ribosomes translate into enzymes. A factory can be constructed by reifying the behaviors defined in Figure 4 as enzymes and placing them inside an object with a type distinct from that of ribosomes:

$$F = \llbracket E_{\mathrm{facA}}, E_{\mathrm{facB}}, E_{\mathrm{facY}}, E_{\mathrm{facZ}}, E_{\mathrm{facZ'}} \rrbracket_1.$$

Behavior *facA* creates a directed bond with any unbonded non-empty object it finds in the factory's neighborhood. This object and its contents serve as the *model*. Behavior *facB* creates a second directed bond from the factory to an empty object with type matching the model. This object serves as the container for the *product*. Behavior *facY* moves behaviors from the neighborhood similar to those in the model into the product. Behavior *facZ* recognizes when the product contains the full set of behaviors and deletes the bond connecting it to the factory. Behavior *facZ′* does the same but also installs a self-directed bond on ribosomes that enables their association behaviors (elements unique to *facZ′* are yellow in Figure 4).

As an initial experiment, we demonstrate mutual replication of a mixed population of ribosomes and factories. Plasmids P_b encoding enzymes E_b comprising ribosomes and factories are placed in a 2D virtual world consisting of 64×64 sites together with a large surplus of ribosomes ($r = 64$) and single instances of factories with ribosome and factory models, F_R and F_F. The supply of primitive combinators and empty objects is replenished as instances are incorporated into enzymes and products. Consequently, the concentration of consumables is held constant. Plasmids and consumables required for synthesis of factory enzymes are overrepresented relative to those for ribosomal enzymes:

$$r\,R + F_R + F_F + \textstyle\sum_B P_b + 2\llbracket\,\rrbracket_0 + 3\llbracket\,\rrbracket_1 + \sum_B \sum_C m_b(c) \llbracket c \rrbracket^-$$

$$\rightarrow (r+1)\,R + 2\,F_R + 2\,F_F + \textstyle\sum_B P_b$$

where the multiset $B = \{\, b \mid E_b \in 2R \cup 3F \,\}$. We observe that the ribosomes synthesize the enzymes encoded by the plasmids and these are then used by the factories to construct additional ribosomes and factories. See Figure 5.

Conclusion

Fifty years after von Neumann described his automaton, it remains a paragon of non-biological life. The rules governing CAs are simple and physical, and partly for this reason, the automaton von Neumann constructed using them is uniquely impressive in its semantic closure. Yet perhaps because RASPs are (in comparison with CAs) relatively well-appointed hosts, self-replicating programs in conventional programming languages seem somehow less convincing. All self-replicating programs must lift themselves up by their own bootstraps, yet not all programs lift themselves the same distance. The field of programming languages has made remarkable advances in the years since von Neumann conceived his automaton. Modern functional programming

[6] Ribosomes without this bond are disabled and serve solely as *models* for *factories*, *i.e.,* as compositional information.

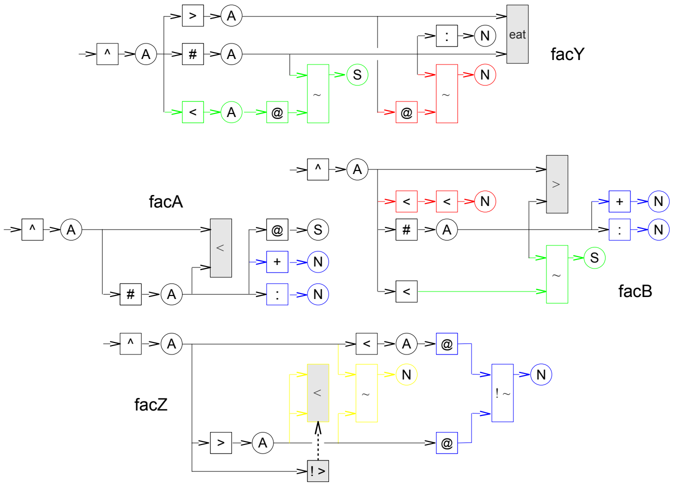


Figure 4: Behaviors defining a factory.

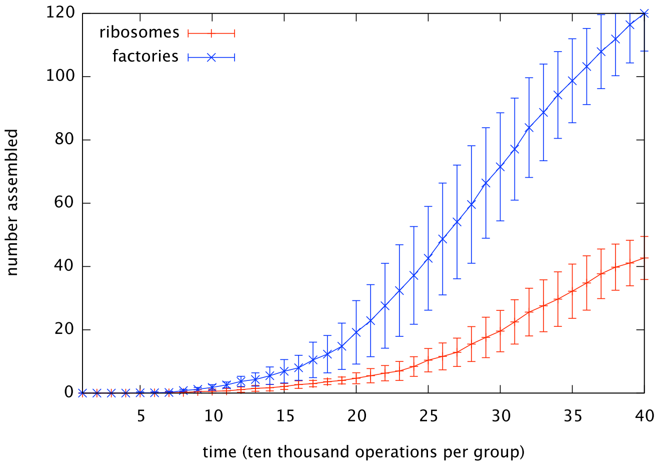


Figure 5: Average increase in numbers of ribosomes and factories (ten runs). Error bars show ± one standard deviation.

languages like Haskell bear little resemblance to the machine languages that are native to RASPs. In this paper, we have attempted to show that programs defined using seemingly exotic constructs like non-deterministic comprehensions can in fact be compiled into sequences of combinators with simple, well-defined semantics. Moreover, because they do not have address operands, these combinators can be reified in a virtual world as actors of only a few fixed types. This makes it possible to build programs that build programs from components delivered by diffusion using processes that resemble chemistry as much as computation.

Acknowledgements

Special thanks to Joe Collard. Thanks also to Dave Ackley, Stephen Harding, Barry McMullin and Darko Stefanovic.

References

Ackley, D. (2013). Bespoke physics for living technology. *Artificial Life*, 34:381–392.

Adami, C., Brown, C. T., and Kellogg, W. (1994). Evolutionary learning in the 2D artificial life system “Avida”. In *Artificial Life IV*, pages 377–381. MIT Press.

Berman, P. and Simon, J. (1988). Investigations of fault-tolerant networks of computers. In *STOC*, pages 66–77.

Berry, G. and Boudol, G. (1990). The chemical abstract machine. In *Proceedings of the 17th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL ’90, pages 81–94, New York, NY, USA. ACM.

Felleisen, M. (1990). On the expressive power of programming languages. In *ESOP’90*, pages 134–151. Springer.

Fontana, W. and Buss, L. W. (1999). What would be conserved if the tape were played twice? In Cowan, G. A., Pines, D., and Meltzer, D., editors, *Complexity*, pages 223–244. Perseus Books, Cambridge, MA, USA.

Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361.

Hutton, T. J. (2004). A functional self-reproducing cell in a two-dimensional artificial chemistry. In *Proc. of the 9th Intl. Conf. on the Simulation and Synthesis of Living Systems (ALIFE9)*, pages 444–449.

Laing, R. A. (1977). Automaton models of reproduction by self-inspection. *Journal of Theoretical Biology*, 66(1):437–456.

McCarthy, J. (1963). A basis for a mathematical theory of computation. In *Computer Programming and Formal Systems*, pages 33–70. North-Holland.

Nakamura, K. (1974). Asynchronous cellular automata and their computational ability. *System Comput. Controls*, 15(5):56–66.

Nehaniv, C. L. (2004). Asynchronous automata networks can emulate any synchronous automata network. *IJAC*, 14(5-6):719–739.

Pattee, H. (1995). Evolving self-reference: Matter, symbols, and semantic closure. *Communication and Cognition - Artificial Intelligence*, 12:9–27.

Paun, G. (1998). Computing with membranes. *Journal of Computer and System Sciences*, 61:108–143.

Ray, T. S. (1994). An evolutionary approach to synthetic biology, Zen and the art of creating life. *Artificial Life*, 1:179–209.

Smith, A., Turney, P. D., and Ewaschuk, R. (2003). Self-replicating machines in continuous space with virtual physics. *Artificial Life*, 9(1):21–40.

Wadler, P. (1990). Comprehending monads. In *Proceedings of the 1990 ACM Conference on LISP and Functional Programming*, LFP ’90, pages 61–78, New York, NY, USA. ACM.

Watson, J. D. and Crick, F. H. (1953). Molecular structure of nucleic acids. *Nature*, 171(4356):737–738.

Williams, L. (2014). Self-replicating distributed virtual machines. In *Proc. of the 14th Intl. Conf. on the Simulation and Synthesis of Living Systems (ALIFE14)*, pages 711–718.

Practical Fault Tolerant 2D Cellular Automata

Steven Janke and Matthew Whitehead

Colorado College
Mathematics and Computer Science
14 E. Cache La Poudre St.
Colorado Springs, CO 80903

{sjanke, matthew.whitehead}@coloradocollege.edu

Abstract

Cellular automata often suffer from a level of brittleness that makes them susceptible to even the smallest unexpected environmental changes. We propose a method of converting CAs into more robust structures called meta-CAs that utilize cell redundancy along with added rules to correct errors and reproduce the functionality of the original CA. We show that the use of these meta-CAs can greatly increase the probability of CAs being intact when executing in an environment where cells fail on each step with a small probability.

Introduction

Cellular automata (CA) are the basis for many examples of computational models in artificial life research. The Game of Life is a traditional CA example where some configurations are mobile, some repeatedly reproduce new subconfigurations, and some simulate logical computation. CAs typically operate by using the neighborhood of a cell to determine the next state of the cell. This means that the CA often suffers from *brittleness* where the loss of a single cell's state completely disrupts the resulting configuration.

For example, Langton's loop (Langton (1984); see Fig.10) was one of the original self-reproducing CAs that consisted of a starting configuration of 86 cells in a particular pattern. During the simulation of the CA, the loop would reproduce yielding an exact replica of itself. If even a single cell of the original pattern is in the wrong initial state, then the entire loop fails to reproduce properly.

This brittleness makes CAs less than optimal for artificial life research since some level of robust behavior is indicative of life-like behavior. In this work, we propose a general procedure for converting a regular CA into a fault-tolerant CA. The procedure involves replacing each original cell with a *metacell* which is a square array of cells with additional states and rules. The new CA is still a normal CA where each cell executes a common set of instructions. The result is a cellular system that mirrors biological cells and possibly could evolve from less robust systems. We focus on two-dimensional CAs and set-up a reasonable interpretation of faulty cells in order to test our procedure on a set of representatives.

Related Work

One of the first to explore fault tolerance in digital systems was von Neumann (1956) who utilized levels of redundancy to build reliable digital circuits from unreliable components. Later in 1975, two sets of researchers, (Harao and Noguchi (1975)) and (Nishio and Kobuchi (1975)), set a more formal stage for studying reliability in cellular automata. In these papers, failure of cells was spatially restricted to make the algebraic analysis feasible.

Theoretical work continued with Peter Gacs' extensive papers (1986, 1989) which include proofs of reliability in certain general settings. More recent work by McCann and Pippenger (2008, 2013) used a majority vote among neighbors to increase the reliability of CAs again in a theoretical setting.

Work in designing cellular automata led to CAs capable of self-reproduction (Byl (1989), Perrier et al. (1996)) and these example CAs are sufficiently complex to serve as test cases for fault tolerant research. One such design (Langton (1984)), referred to as Langton's loop, helped motivate this current research and serves as a key example in our original testing.

Work using self-reproducing loops took a more biological tact when Hiroki Sayama described loops that were able to dissolve when intruded upon (Sayama (1998)). His work continued with loops that evolved over time based on environmental constraints (typically lack of space) (Sayama (1999)). Sayama later described CAs that were able to employ defensive mechanisms to remain intact when facing intrusions (Sayama (2004)). Oros et al. more recently describe a system that uses loops that are able to reproduce and pass along copies of the parents' genetic material (Oros and Nehaniv (2007)).

Cellular Automata and Reliability

Design

Two-dimensional cellular automata which are the focus of this research are rectangular arrays of identical finite automata. An individual finite automaton is called a *cell* and the eight cells surrounding a given cell in the array comprise the *neighborhood*. Traditionally, there are two interpretations of the neighborhood. In the von Neumann design, the neighborhood is just the four cells bordering to the north, east, south, and west; the other four are ignored. The Moore design includes all eight cells in the neighborhood.

The cells operate on discrete time steps and in one step, a cell uses its current state plus the state of the neighbors to determine its next state. The transition is described by a function that sends the neighborhood configuration along with the current state of the central cell to a new state for the central cell. This transition function can be presented as a set of rules. For a Moore neighborhood design, each rule is a 10-tuple of states, $(S_{cur}, S_N, S_{NE}, S_E, S_{SE}, S_S, S_{SW}, S_W, S_{NW}, S_{new})$. Here, the given cell is currently in state S_{cur} and the eight neighbors are in the states listed. On the next time step, the given cell transitions to state S_{new}. This means that for the Game of Life CA, since there are two cell states (usually designated "alive" and "dead"), there are 2^8 possible Moore neighborhoods. The given cell has two possible states S_{curr}, so there are $2^8 \times 2 = 2^9$ rules in the transition function. In practical implementations, many of these rules are inactive in the sense that they do not change the state of the central cell; utilizing this observation can speed up the actual simulation. Also, it is generally agreed that state 0 is designated as a quiescent state and therefore the tuple (0,0,0,0,0,0,0,0,0,0) is a rule in most CAs.

In general, with S states, there are S^9 rules for a Moore neighborhood CA and S^5 rules for a von Neumann neighborhood CA. (The total number of transition functions and hence CAs with S states using the Moore neighborhood is then S^{S^9}.) The number of activated rules along with the number of states gives a rough measure of complexity for the CA. One constraint in converting a CA to a more fault-tolerant form is the level of complexity introduced. Some redundancy has to be introduced to improve reliability, but it is also useful and important to give some consideration to the minimal necessary redundancy.

The rules determine a CA, but in order for the CA to do useful work or display useful behavior, we also need an *initial configuration*. This is simply a finite set of cells containing at least one non-quiescent state. The initial configuration and the set of rules together determine the fault-tolerance of a CA. For example, in the Game of Life the set of rules (using the Moore neighborhood) basically quantifies the two intuitive rules:

1. Any "dead" cell becomes alive if it has exactly 3 "live" neighbors.
2. Any "live" cell stays alive only if it has 2 or 3 "live" neighbors.

With these rules, an initial configuration forming a 2×2 block of live cells, is fault-tolerant in the sense that if any of the four become dead, the configuration repairs itself. On the other hand, a *glider* configuration, which will move diagonally across the cellular array in 4 steps, is not fault-tolerant. If any of the cells die, the diagonal movement fails. Cells which are particularly critical for the continued successful operation of the CA are referred to as *essential cells*.

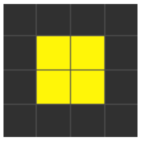


Figure 1: A simple block in the Game of Life that repairs the loss of any single one of its cells.

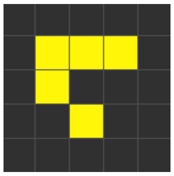


Figure 2: A glider in the Game of Life fails to continue its diagonal movement pattern when any of its cells die.

Cell Lifetime

In order to specify the source of faults in CAs, we use machine lifetime as an analogy. Each cell is a finite automaton with a finite lifespan. At the end of a cell's lifetime, it is immediately replaced and the new cell begins in the quiescent state. Probabilistically, if a cell has probability p of failing on any given time step and if we assume each time step is independent (more theoretical than practical), then the expected lifetime of the cell is $1/p$. We assume that all cells in an array have the same lifetime. On each step of the CA simulation, each cell has probability p (independently) of failing and hence needing replacement. We refer to this as a $p - death$ environment.

Note that there are many ways of introducing faults into a CA. A fault might mean that a cell spontaneously transitions to another state rather than to the quiescent state as in the lifetime model. (Notice that the 2×2 block in the Game of Life CA is not tolerant to spontaneous transitions to active states.) In some previous work, the fault model assumes that there is a maximum number of faults in each spatial region of the CA. Yet another model might allow periods of no faults giving the CA time to repair. However, in this study we settle on the lifetime model and allow each cell to independently experience a fault with some common probability.

Test Cases

To test a CA for fault tolerance, we first identify a specific task that the CA should complete. For example, we might decide that the key function of the Langton loop CA is to produce one additional copy of itself which it may do in n steps. So we run the CA for n steps in a $p - death$ environment and then remove the threat of failures for an additional k steps. If the CA effectively repairs itself, it is counted as a success. The proportion of successes is then recorded as the reliability or fault tolerance.

Note that there are three sources of ambiguity here. First, it is not always obvious when a CA has completed a task; for a self-reproducing CA, it may be obvious, but for a digital gate simulation, perhaps only the correct signal needs to pass through the gate. Second, the number of extra steps k is not well-defined; in some test cases we took $k = 0$ and in others we used $k = n$. Finally, in practical uses of CAs, it is not necessary for the final state to exactly match the CA with no faults. In other words, it may only be a sub-configuration, like the digital signal through a gate, that needs to be correct.

To test our procedure for constructing fault-tolerant CAs, we focused on the following CAs with von Neumann neighborhoods (see `http://cs.coloradocollege.edu/~mwhitehead/metacells` for .RLE and .table files).

- Single Transition (Figure 3): This is the most primitive CA since it simply starts with a configuration of five cells and transitions (using the rules of any CA) to the new state for the central cell. Testing this configuration gives us information for predicting how larger CAs may perform. Experiments are considered successful for this CA if the center cell is correct after the single transition.

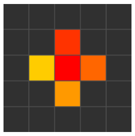


Figure 3: Single transition structure

- Glider (Figure 4): Unlike the glider configuration in the Game of Life which uses a Moore neighborhood, this glider functions with a von Neumann neighborhood and three rather than two states. Figure 4 shows the stages in moving from right to left. Experiments are considered successful for this CA if the glider is intact after flying to the left for four simulation steps.

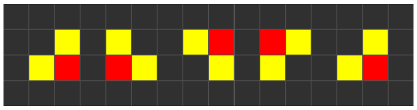


Figure 4: Glider Stages: moves from right to left

- Coral (Figure 5): This is a slightly more complicated CA whose starting configuration is simple, but it produces a symmetric structure that continually grows in size (Figure 5 also shows the structure after eight steps). Experiments are considered successful for this CA if the coral structure is intact after eight simulation steps.

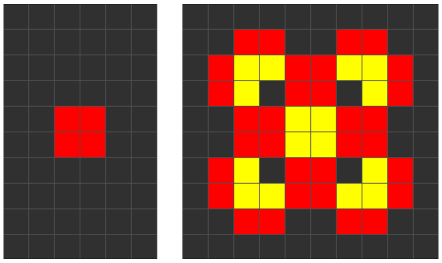


Figure 5: Coral: Start(left) and after 8 steps (right)

- XOR Gate (Figure 6): This CA simulates the digital XOR gate with two inputs and one output. The final configuration after 14 steps is shown in Figure 7. Experiments are considered successful if the correct signal is sent out to the end of the structure and the gate is intact.

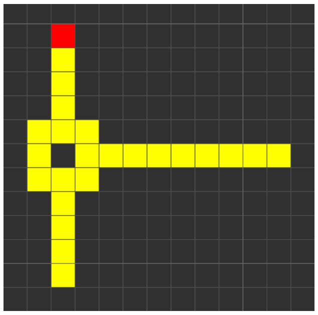


Figure 6: XOR Gate Starting Structure – Input 1 at top and input 0 at bottom

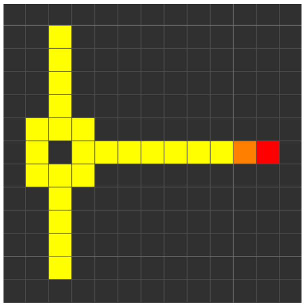


Figure 7: XOR Gate after 14 steps

- Byl's Loop (Figure 8): One of the key self-reproducing CAs. After 25 steps, the evolving configuration contains two copies of the initial configuration (Figure 9). Experiments are considered successful for this CA if the loop can correctly copy itself one time.

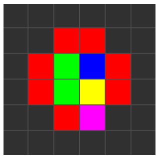


Figure 8: Byl's loop starting structure

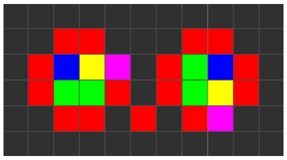


Figure 9: Byl's loop after one successful reproduction (25 steps)

- Langton's Loop (Figures10, 11): This is a larger configuration that reproduces after 151 steps. As with Byl's loop, experiments with Langton's loop are considered successful one additional copy of the loop is produced correctly.

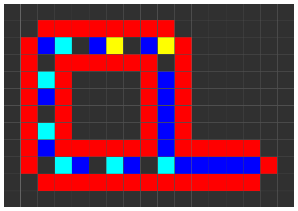


Figure 10: Langton's loop starting structure

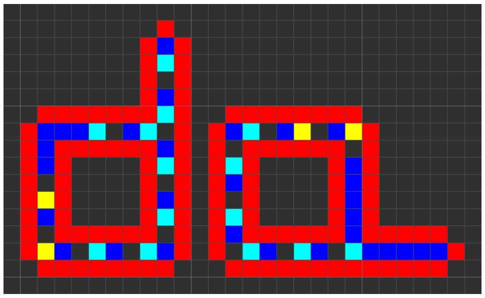


Figure 11: Langton's loop after one successful reproduction (151 steps)

Metacells

Our procedure for converting CAs into reliable systems relies heavily on the concept of a *metacell*. These are square arrays of cells that replace each of the single cells in the original CA. The point is to provide informational redundancy to improve fault tolerance, but in order to do so, the rules governing transitions in the original CA have to be altered to now simulate the original operations on square arrays of cells. One constraint is that the converted CA must still be a genuine CA; each cell in the conversion follows a common set of rules. The transformation to a converted CA (called a meta-CA), theoretically works on either Moore or von Neumann neighborhoods, but it is much more straightforward for original CAs using the von Neumann neighborhood.

There are several ways to design a metacell and our approach evolved as we tried to balance the achieved redundancy with the number of states in the CA. We set an arbitrary threshold of 256 states since that is the limit of the Golly simulator (Trevorrow and Rokicki (2013)). This practical limit did help focus our work on the efficiency of metacell design, but it is clear that using a larger simulator has its advantages. For more complex test cases, we had to approximate how successful the tolerant meta-CAs would be.

Figure 12 shows an example metacell of size 3. The 3x3 grid in the interior (yellow cells) of the metacell stores the redundant information that collectively represents a state from the original CA cell. A common flood-fill algorithm can keep the state information intact despite random faults. The *border cells* (red cells in the figure) serve as a boundary between metacells and can also be used as a built-in clock to synchronize the operation of the meta-CA. Meta-CAs operate at a slower speed than their normal, single-cell CA equivalents. The application of a single rule in an original CA is translated to several steps in a meta-CA. These additional steps are required in order to allow metacells to recover information disrupted by cell faults and to propagate state information from the neighboring metacells.

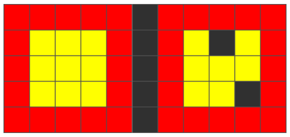


Figure 12: Bordered metacell of size 3 (left) and bordered metacell with two dead interior cells (right)

The weakness of this first metacell design is that there are several *essential* cells. Both the border cells and the signals carrying state information into the interior of a metacell have no redundancy. The success of this design relies on keeping the ratio of essential cells to total cells rather small.

In order to reduce the essential cells, we altered the flood-fill algorithm to simulate mixing paint. The states of neighboring cells are considered paint colors that flow into a metacell. Whenever two such colors are neighbors, the corresponding cells transition to a state representing the mix of the two colors. Similarly, whenever two mixed colors meet or a mixed color and a third plain color meet, there is a transition to a cell representing a mix of all colors involved. This ensured that there was not just a single cell carrying the message on any clock step. The number of essential cells was decreased and the success of the meta-CA increased. However, the boundary cells remained and consequently there were still vulnerable cells (not necessarily essential) in the meta-CA. The mixed-paint principle unfortunately requires on the order of n^4 additional states where n is the number of states in the original CA.

Positional Metacells

Eliminating the border cells improves fault tolerance and our final approach, which we call *positional metacells*, was the most successful. Positional metacells encode a cell's position in the metacell along with the original state information. A cell would then have information showing that it was in the upper-left corner of the metacell, for example. Positional metacells use Moore neighborhoods to simulate an original CA that used von Neumann neighborhoods. Once again, more states are needed to cope with the positional information. Nevertheless, the positional metacell design does lead to improved fault tolerance since failed cells can be repaired based on their relative position in relation to other correct cells.

The simplest kind of positional metacell is a 2x2 grid as shown in Figure 13. Each of the four cells comprising the metacell contains information about its location within the metacell (signified by the numbers in the figure) and the state value from the original CA.

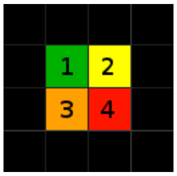


Figure 13: A 2x2 Positional Metacell. Each interior cell is encoded by its position within the metacell.

The metacell transitions take two steps. On the first step, each metacell interior cell changes to a state that represents the combination of its two neighbors belonging to other adjacent metacells. This is done using the color-mixing idea from above. Note that there is redundancy in each of the incoming state messages. For example, the upper-left cell changes to a state representing the combination of the metacell's top and left metacell neighbors and the upper-right cell encodes the metacell's top and right neighbors. This means that the top metacell neighbor's value is encoded in two places. This helps eliminate any essential cells and provides a more reliable structure.

On the second step, the original CA's transition is computed using the four mixed color values. Because of the redundancy of the encoded neighbor information, if any single cell dies, then the original transition can still be calculated. In fact, as long as two cells that are diagonally opposite one another are intact, then the original transition can be performed. The output of the original transition is then positionally encoded throughout the four interior cells and one metacell transition is complete.

These metacells do not need separate border states because of the positional encodings, but the positional encodings do require four states per original CA state. They also require states to represent all combinations of mixes of three original states since each positional cell must encode its own value along with the two neighbor metacell cells that it borders. The final number of required states is $4n^3 + 4n + 1$, where n is the number of original states.

Because of their lack of essential cells and short clock cycle of two steps, these 2x2 positional metacells were the most fault tolerant design we experimented with. The results for the various test cases reported below compare the reliability of these structures versus their original CA counterparts.

Converting Existing CAs to Positional Metacell CAs

One of the appeals of using positional metacells for fault tolerance is that existing CAs can be converted automatically into their meta-CA equivalents. We have developed a set of conversion scripts to perform this process. (See `http://cs.coloradocollege.edu/~mwhitehead/metacells` for scripts.)

The first part of the conversion process involves changing the original CA structure into a meta-CA structure using the appropriate new state values. For example in the 2x2 positional metacell case, a single cell with state value 3 from the original CA must be converted into four cells that are positionally encoded as *original state 3 in the upper-left corner, original state 3 in the upper-right corner, original state 3 in the lower-left corner, and original state 3 in the lower-right corner*. Figure 14 shows a simple 3-cell CA converted to the corresponding meta-CA.

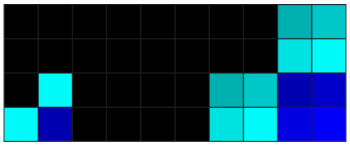


Figure 14: Original CA (left) and Positional Meta-CA (right)

Once the structure of the original CA has been converted, then the original ruleset needs to be converted to include the rules for meta-CA operation. Meta-CA operation requires several different types of rules. In particular, rules must be included to encode the combination of neighbor states, ignore and repair missing states (if possible), and perform the original CA's transitions.

First, rules must be added to convert each positionally-encoded cell to a new state representing the combination of its two neighbors from adjacent metacells. For example, the

A cell in Figure 15 needs a rule to transition to the state that represents the combination of starting state A, X above, and Y to the left. Similar transition rules are added for each interior cell position and all combinations of possible external neighbor states.

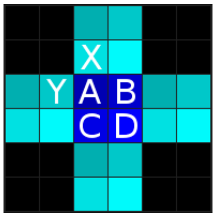


Figure 15: A, B, C, and D form one metacell. A's external neighbors are X and Y and its internal neighbors are B, C, and D.

Second, rules must be generated to ignore dead cells when possible while still performing the correct transitions. Again using Figure 15 as an example, if cell A were dead it can still properly transition to the state that encodes the combination of X and Y as long as one of its internal neighbors is intact. So if A is dead, but B is intact, then cell A can correctly transition to the combination of X and Y since B *is on its right*. This is the added fault tolerance of using the positional encoding scheme.

Finally, rules are added to perform the original CA's transitions. These transitions occur on the second step of the meta-CA's cycle. Suppose the original CA had the rule (1,2,3,2,4,5). That is, if a cell were in state 1 and had neighbors starting at the top and going clockwise of 2, 3, 2, and 4, then it should change to state 5. Using the labels in Figure 15, the cells would have the following mixes after the first step of the meta-CA's simulation. Cell A would have a mix of the original state 2 from above and state 4 from the left, cell B would have a mix of state 2 from above and state 3 from the right, cell C would have a mix of state 2 from below and state 4 from the left, and cell D would have a mix of state 2 from below and state 3 from the right.

These cells then need to transition to the positionally-encoded state that corresponds to the original CA's new state after the transition. For this example, A would need to transition to the state representing the *original state 5 in the upper-left corner*. In order to perform these kinds of transitions, rules must be included to decode the mixes of states in the context of the original rules. For example, cell A has information about the states from above and left and cell D has information about the states from below and right, so cell A gets a rule that transitions to its new state (*original state 5 in the upper-left corner*) when it has D's state as a lower-right neighbor. Similar rules are added for the other positions in the metacell and for all other possible internal neighbor states.

Notice again that external neighbor state information is redundantly encoded. For example, both cells A and B encode the original state 2 from above. If at least one of these two cells is intact, then the neighbor information from above is not lost and the original transition can be successfully performed.

Results

Using the test cases mentioned above, we ran a series of experiments to test the fault tolerance of our proposed meta-CAs against normal CAs. The results listed here are for the 2x2 positional meta-CAs, as they were the most successful model that we tested. We used a range of p-death environments and ran 1000 trials per test case. For larger structures that require too many states for the Golly simulator, we list predicted success rates calculated by using the experimental success rates for single cell transitions raised to the power of the number of required transitions for the given CA. To ensure accurate single cell transition success rates, we performed 200,000 single cell transition trials.

p	Original	Meta-CA
0.00001	99.9980	100.0000
0.00002	99.9960	100.0000
0.00005	99.9900	100.0000
0.00010	99.9800	100.0000
0.00020	99.9600	100.0000
0.00050	99.9000	99.9999
0.00100	99.8001	99.9996
0.00200	99.6004	99.9984
0.00500	99.0025	99.9901
0.01000	98.0100	99.9604
0.02000	96.0400	99.8432
0.05000	95.0250	99.0494
0.10000	81.0000	96.3900

Table 1: Success rates for a single cell/metacell transition.

In Table 1, the Meta-CA success rates were calculated using the formula $1-(p^4+4p^3(1-p)+4p^2(1-p)^2)$. This follows from the binomial distribution where p^4 is the probability of all four cells being dead, $4p^3(1-p)$ is the probability of exactly three cells being dead (there are four ways for this to occur) and $4p^2(1-p)^2$ is the probability of two non-diagonal cells being dead (there are also four ways for this to occur). A simulator experiment confirmed these results.

The tables of results list the percentage of trials that were successful, with success defined as above for each test case. The results show that meta-CAs provide an added level of fault tolerance across a variety of CA structures and levels of p-death environments. In general, the greater the number of cells and required transitions in the CA structure, the greater the benefit of using a meta-CA representation. This is to be expected since metacells outperform regular CA cells for each individual cell transition.

The type of p-death environment makes a difference in the degree of added fault tolerance provided by meta-CAs. For very low values of p, both regular CAs and meta-CAs are likely to be successful. On the other hand, very high values of p usually cause both regular CAs and meta-CAs to fail. Intermediate p values show the greatest benefit of using meta-CAs. For example, using the coral-like structure with $p = 0.005$, the meta-CA was successful 97.7% of the time while the normal CA was only successful 50.3% of the time.

p	Original	Meta-CA
0.00001	100.0	100.0
0.00002	100.0	100.0
0.00005	100.0	100.0
0.00010	100.0	100.0
0.00020	100.0	100.0
0.00050	99.5	100.0
0.00100	99.4	100.0
0.00200	98.5	100.0
0.00500	97.1	99.8
0.01000	92.0	99.1
0.02000	85.6	97.5
0.05000	71.6	86.9
0.10000	41.0	52.0

Table 2: Success rates for a von Neumann neighborhood glider to complete one cycle in 4 steps.

p	Original	Meta-CA
0.00001	100.0	100.0
0.00002	99.5	100.0
0.00005	99.2	100.0
0.00010	98.3	100.0
0.00020	97.1	100.0
0.00050	93.2	100.0
0.00100	85.3	99.7
0.00200	76.8	99.4
0.00500	50.3	97.7
0.01000	25.4	89.2
0.02000	5.5	63.0
0.05000	0.1	5.8
0.10000	0.0	0.0

Table 3: Success rates for a von Neumann neighborhood coral to be intact after 8 steps.

Complexity of Meta-CAs

All of our constructions centered on duplicating information in the original CA. The most straightforward approach is to introduce more cells for redundancy as is done in a metacell. However, in order to benefit from the redundancy there must be a mechanism for combining information in order to recover from information lost in faulty cells. For example, border cells were used to keep information within the bounds of a particular metacell and signals were transmitted

p	Original	Meta-CA (predicted)
0.00001	99.6	100.0
0.00002	99.2	100.0
0.00005	98.0	100.0
0.00010	97.0	100.0
0.00020	93.6	100.0
0.00050	84.5	99.7
0.00100	72.9	99.7
0.00200	48.4	99.5
0.00500	18.8	97.0
0.01000	2.7	88.6
0.02000	0.1	59.4
0.05000	0.0	4.0
0.10000	0.0	0.0

Table 4: Success rates for a von Neumann neighborhood XOR gate to produce the correct output value.

p	Original	Meta-CA (predicted)
0.00001	99.1	100.0
0.00002	98.7	100.0
0.00005	97.8	100.0
0.00010	95.0	100.0
0.00020	91.1	100.0
0.00050	76.2	99.5
0.00100	57.0	99.5
0.00200	36.2	99.2
0.00500	6.6	95.3
0.01000	0.5	82.6
0.02000	0.0	43.8
0.05000	0.0	0.6
0.10000	0.0	0.0

Table 5: Success rates for a Byl loop to reproduce once.

p	Original	Meta-CA (predicted)
0.00001	84.3	100.0
0.00002	70.5	100.0
0.00005	43.9	100.0
0.00010	17.3	100.0
0.00020	3.4	100.0
0.00050	0.0	83.1
0.00100	0.0	83.1
0.00200	0.0	75.7
0.00500	0.0	18.9
0.01000	0.0	0.1
0.02000	0.0	0.0
0.05000	0.0	0.0
0.10000	0.0	0.0

Table 6: Success rates for a Langton loop to reproduce once.

between metacells. Since cells in a CA are homogeneous, there cannot be border cells or signals without additional states. This suggests that some measure of CA complexity based on the number of states might determine the minimal number of additional states necessary to achieve a given level of fault tolerance.

A recent paper (Lui et al. (2015)) describes work that uses both the Shannon measure of complexity (based on entropy) and the Kolmogorov measure (based on the length of code description) to construct a hybrid measure that shows some promise in distinguishing one dimensional cellular automata. This measure takes into account both the CA rules and the starting configuration. In our work, it is not yet clear that there is a direct relationship between the starting configuration and the level of achieved redundancy. Clearly, some configurations are more brittle than others, but our procedure for introducing redundancy is independent of the starting configurations and it may be that more simple measures of the CAs structural complexity could shed light on the number of additional states necessary to add appropriate redundancy. Currently, we plan to investigate the effect of active rules (those that change the central cell state) on the fault tolerance of our meta-CAs.

Conclusions

This work offers evidence for the following CA design principles:

1. A practical general purpose algorithm can convert an original CA into a more fault-tolerant meta-CA. The definition of fault-tolerance in this context refers to cells with a lifetime that is geometrically distributed.

2. The general design of meta-CAs involves the inclusion of metacells, and this research identified one promising metacell type: positional metacells. First attempts to design metacells required too many essential cells in mechanisms to manage the redundancy. The fault tolerance did not scale well to higher amounts of redundancy. Our positional metacells proved to be the most efficient (number of states).

3. Complexity of the derived metacell is an issue. The amount of redundancy and the way it is managed introduces many new states to the original CA. Positional metacells kept the number of additional states within reason, but we hypothesize that there may be hopefully simple measures of complexity that might lead to bounds on the number of additional states necessary to reach conclusions about the minimum number of additional states.

Acknowledgements: The authors wish to thank the reviewers for their useful suggestions.

References

Byl, J. (1989). Self-reproduction in small cellular automata. *Physica D*, 34:295–299.

Gacs, P. (1986). Reliable computation with cellular automata. *Journal of Computer and System Science*, 32:15–78.

Gacs, P. (1989). Self-correcting two-dimensional arrays. *Advances in Computing Research*, 5:223–326.

Harao, M. and Noguchi, S. (1975). Fault tolerant cellular automata. *Journal of Computer and System Sciences*, 11:171–185.

Langton, C. G. (1984). Self-reproduction in cellular automata. *Physica D*, 10:135–144.

Lui, L., Terrazas, G., Zenil, H., Alexander, C., and Krasnogor, N. (2015). Complexity measurment based on information theory and kolmogorov complexity. *Artificial Life*, 21. (to appear).

McCann, M. and Pippenger, N. (2008). Fault tolerance in cellular automata at high fault rates. *Journal of Computer and System Sciences*, 74:910–918.

McCann, M. and Pippenger, N. (2013). Fault tolerance in cellular automata at low fault rates. *Journal of Computer and System Sciences*, 79:1136–1143.

Nishio, H. and Kobuchi, Y. (1975). Fault tolerant cellular spaces. *Journal of Computer and System Sciences*, 11:150–170.

Oros, N. and Nehaniv, C. L. (2007). Sexyloop: Self-reproduction, evolution and sex in cellular automata. In *The First IEEE Symposium on Artificial Life*, pages 130–138.

Perrier, J.-Y., Sipper, M., and Zahnd, J. (1996). Toward a viable, self-reproducing universal computer. *Phys. D*, 97(4):335–352.

Sayama, H. (1998). Introduction of structural dissolution into Langton's self-reproducing loop. In *Proceedings of the 6th International Conference on Artificial Life (ALIFE-98)*, pages 114–122, Cambridge, MA, USA. MIT Press.

Sayama, H. (1999). Toward the realization of an evolving ecosystem on cellular automata. In *In M. Sugisaka and H. Tanaka (Eds.), Proceedings of the Fourth International Symposium on Artificial Life and Robotics (AROB 4th 99)*, pages 254–257.

Sayama, H. (2004). Self-protection and diversity in self-replicating cellular automata. *Artificial Life*, 10(1):83–98.

Trevorrow, A. and Rokicki, T. (2013). Golly cellular automata software. `http://golly.sourceforge.net/`. Version 2.6, 2013.

von Neumann, J. (1956). Probabilistic logics and the synthesis of reliable organisms from unreliable components. In *Automata Studies (ed. C.E.Shannon, J.McCarthy)*, pages 43–98. Princeton University Press.

Evaluating the Effect of a Flexible Spine on the Evolution of Quadrupedal Gaits

Jared M. Moore[1], Craig P. McGowan[2], and Philip K. McKinley[1]

[1]Dept. Computer Science and Engineering, Michigan State University, East Lansing, Michigan, USA
[2]Dept. Biological Sciences, University of Idaho, Moscow, Idaho, USA
moore112@msu.edu

Abstract

Animals demonstrate a level of agility currently unmatched in their robotic counterparts. The elasticity of muscles and tendons increase not only performance, but also the efficiency of movements. In contrast, robots are often constructed with rigid components connected by motors. However, recently compliant actuators and materials have been introduced to enhance robot designs, emulating the flexibility of natural organisms. In this paper, we incorporate passive flexibility into the spine of a quadruped animat and employ computational evolution to generate gaits. Results indicate that spine flexibility significantly increases both performance and efficiency of evolved individuals. Moreover, evolving the degree of spine flexibility along with artificial neural network controllers produces the highest performing solutions.

Introduction

Animals exhibit a diversity of behaviors that allow them to survive in dynamic environments, while having simultaneously evolved to be energetically efficient (Cavagna et al., 1977). Muscles perform the majority of work, but they are supported by the underlying skeletal and tendon systems. The inherent elasticity of muscles and tendons contributes to efficiency of movement throughout an animal's stride (Alexander, 1984). Beyond providing the basis for posture, the spine plays an important role in the energy efficiency of movements (Alexander, 1988). In particular, the spine can adopt very different roles. For example, in galloping horses, the spine acts similar to a stiff spring with minimal flexibility, while in sprinting cheetahs the spine moves actively, lengthening the stride (Hildebrand, 1959).

In contrast to their biological counterparts, robotic systems are considerably less flexible, typically comprising rigid components interconnected by motors. The addition of structures such as actuated spines can enhance the functionality of legged robots by increasing movement freedom (Leeser, 1996). Specifically, actuated spines expand the range of possible gaits (Berns et al., 1998) and increase maneuverability in constrained spaces (Park and Lee, 2007). In simulation, actuated spines have been shown to increase both the speed and efficiency of bounding gaits in a 2D quadruped, consistent with observations in biomechanics (Culha and Saranli, 2011).

Another possible approach to increasing the movement freedom of robotic systems is to incorporate passive compliance. Compliant designs exploit the intrinsic properties of materials, emulating the flexibility inherent in biological systems. Recently, other investigations have examined the integration of flexible materials into morphological components such as fish fins (Clark et al., 2012; Epstein et al., 2006; Clark et al., 2014), compliant actuators (Van Ham et al., 2009), and flexible arms (Sfakiotakis et al., 2014). Silva et al. (2005) described quadruped locomotion for a simulated animat with a flexible spine, but did not quantify the effect of that component on performance.

In this paper, we examine how a passively flexible spine influences the *evolution* of locomotive behaviors in a quadruped animat with an artificial neural network (ANN) controller. We first evolve gaits for an animat with a rigid spine. Next, we create a three-segmented spine with predefined flexibility. Finally, we allow the degree of flexibility to evolve along with the ANN controller. Results indicate that adding passive flexibility to a quadruped can significantly increase performance and efficiency of movement. These results complement earlier investigations in actively controlled spines, demonstrating that passive flexibility is beneficial, despite requiring no direct control from the ANN. Apparently, the additional degrees-of-freedom (DOF) allow the evolved controllers to express gaits that are not possible with a rigid spine. These results have practical implications for robotic systems by improving performance as well as efficiency.

Related Work

Incorporating flexible materials holds promise for increasing the functionality of robotic systems. In aquatic robots, mimicking fin-like appendages can increase agility, power efficiency, and robustness. Anderson and Chhabra (2002) demonstrated a hybrid rigid-body, flexible-tail robotic tuna that exhibited increased maneuverability at high speeds compared to traditional autonomous underwater vehicles (AUV). In addition, Krishnamurthy et al. (2010) presented an AUV design based on the morphology of an electric ray that employs flexible materials in the tail to increase efficiency. Flexibility has also been explored in terrestrial robotics. For example, modeling the flexibility of biological muscles has produced life-like bipedal gaits for simulated robots by allowing the body to absorb shock when contacting the ground (Geijtenbeek et al., 2013). Further, Ackerman and Seipel (2013) have shown that reducing the vertical movement of the center of mass during walking reduces energy consumption in legged robots. This reduction is similar to the dampening effect provided by tendons in biological organisms, reducing collision forces with the ground and increasing efficiency (Bertram and Hasaneini, 2013; Ruina et al., 2005). However, controlling flexible joints can be challenging as they react less predictably to controller input and interactions with the environment (Chaoui et al., 2009).

One possible approach to addressing the control problem is through evolutionary robotics (Sims, 1994; Nolfi and Floreano, 2000), where the controller evolves to take advantage of morphological characteristics. Evolutionary approaches have produced effective robotic gaits in quadrupeds (Clune et al., 2009; Doncieux and Mouret, 2013), salamanders (Ijspeert et al., 2005), and bipeds (Lessin et al., 2013). In addition to evolving active control strategies, co-evolving morphology and control can exploit relationships between brain and body (Bongard, 2011; Paul, 2006; Valsalam and Miikkulainen, 2008; Rieffel et al., 2010). Furthermore, computational evolution has proven effective at exploiting passive flexible materials (Moore and McKinley, 2012) and passive joints (Moore and McKinley, 2013) in robotic systems. In this paper, we evolve control and spinal flexibility in quadruped animats.

Methods

Simulation Environment Simulations are conducted with the Open Dynamics Engine (ODE) (Smith, 2013), a 3D physics simulation environment. Although designed to simulate rigid bodies, flexible components can be modeled by interconnecting multiple segments with spring-like joints. ODE also handles collisions between 3D bodies and forces such as friction and gravity. In this study, the environment is a flat, high-friction surface minimizing slippage.

Evolutionary runs are conducted with two quadruped animats. The first animat has a single, rigid torso with four legs. The second, shown in Figure 1, has a three segment torso connected by passively flexible, 2-DOF joints. This configuration emulates the flexibility of a spine in a natural organism, allowing for increased movement freedom over the single, rigid torso quadruped. The legs of both quadrupeds have two 2-DOF joints, a hip and knee, for a total of eight actively controlled joints. Movement of the legs can be away from, or along the long axis of the body.

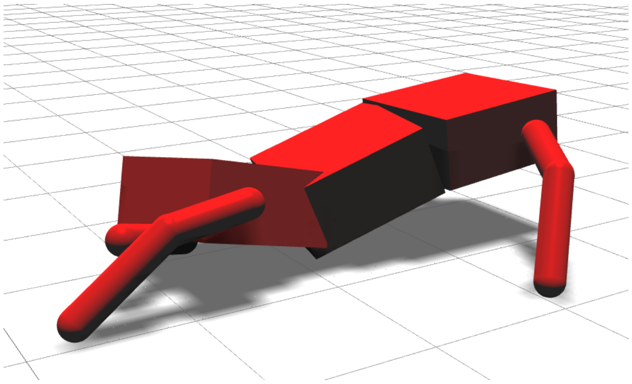


Figure 1: Quadruped animat with a flexible spine. The torso is divided into three segments connected by passively flexible joints which move in reaction to external forces that arise from leg movement.

Treatments We conduct three treatments. The first (*Rigid-Spine*) uses the single, rigid-torso quadruped described above. The second treatment (*Flex-Spine*) has a quadruped with a three-segmented torso and a predefined (passive) flexibility. The third treatment (*Evo-Flex*) also has a three-segmented torso, but the level of flexibility in the passive spine evolves with the controller.

Passive Flexibility As noted above, we model a flexible spine by connecting rigid boxes with spring-like joints. When flexed, a joint attempts to return its two connected bodies back to their neutral position. Together, the three segments and two spring joints allow the torso to flex side-to-side and up/down. Both joints in the animat have the same level of flexibility. Thus, the torso can contribute to movement, rather than serving a purely structural purpose. In the *Evo-Flex* treatment, spinal flexibility can range from very light springs (low spring coefficients), enabling the spine to move easily, to very stiff springs (high spring coefficients), providing minimal flexibility of the spine and returning the torso segments more rapidly to their neutral position.

Artificial Neural Network Controllers are evolved with the NEAT algorithm (Stanley and Miikkulainen, 2002), which uses a genetic algorithm to evolve recurrent artificial neural networks (ANNs). NEAT begins with a fully connected input-to-output network without hidden nodes, complexifying the ANN by adding nodes and links over evolutionary time. Speciation addresses the issue of two dissimilar network topologies acting as parents for an individual by determining compatibility between networks, thus defining which ANNs can be crossed over. A population of 120 individuals is evolved for 4000 generations. NEAT parameters include: max species = 25, mutation rate = 0.33, add neuron probability = 0.4, add link probability = 0.4, and remove link probability = 0.05. ANNs have 22 inputs: a pe-

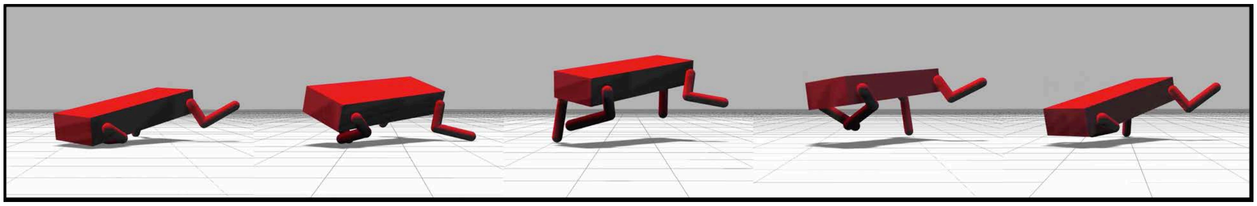


Figure 2: An evolved quadruped gait with a single, rigid torso. This is a rear bounding gait taken from one of the highest performing individuals from the *Rigid-Spine* treatment.

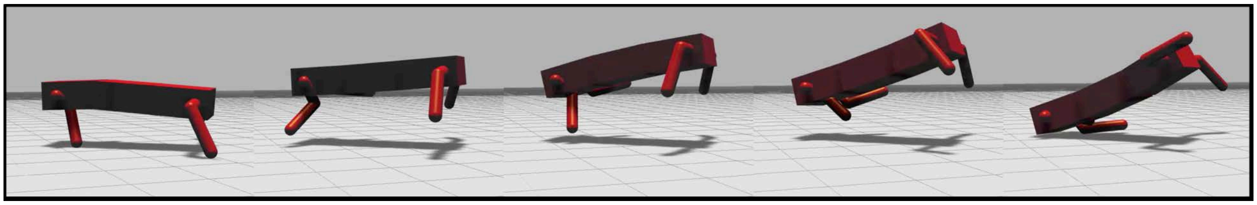


Figure 3: Evolved quadruped gait with a three-segmented torso connected by passively flexible joints from the *Flex-Spine* treatment. The spine flexes throughout the gait providing increased compliance in the robot resulting in higher performance than the *Rigid-Spine* treatment.

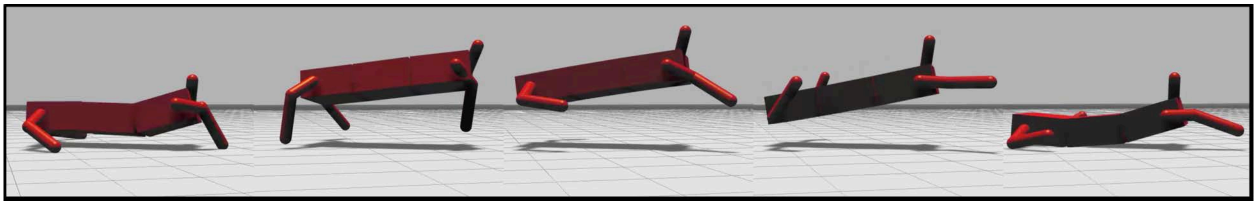


Figure 4: An evolved gait from the *Evo-Flex* treatment. Here, the spine is highly flexible, contributing to the gait by lengthening the stride while the legs are in contact with the ground.

riodic oscillating signal (1), two joint angle sensors per joint (16), touch sensors on each foot (4), and a bias (1). The 16 outputs specify the desired angles of the hinge joints, two per each 2-DOF joint.

Fitness and Efficiency Evaluation Individuals are evaluated based on the Euclidean distance from their initial position to the final position after 10 seconds of simulation time. In order to reduce the bias on the evolutionary process, we do not constrain movement to a particular direction. During an evaluation, we record the forces exerted by each joint to determine an individual's efficiency. We calculate efficiency as the distance traveled per unit of power exerted by a robot. In previous work (Moore and McKinley, 2015), we found that evolved quadrupeds exhibit an inherent level of efficiency during movement, even when selecting for performance only.

Experiments & Results

Several distinct gaits evolved across the three treatments. They can be classified as bounding (rear legs provide power, front legs stability), shuffling (low posture, body often on the ground), and trotting (diagonally paired gait). Figures 2, 3, and 4, respectively, illustrate sample gaits from each of the three treatments. High performing gaits evolve in every treatment. Videos of selected gaits from the three treatments are available at the following addresses:

Rigid-Spine: `http://youtu.be/laChVR5LWgE`
Flex-Spine: `http://youtu.be/AynOi6tBg0A`
Evo-Flex: `http://youtu.be/oW-tLQx5DSc`

Rigid Spine We first examine the evolution of quadrupedal gaits in the *Rigid-Spine* treatment. Each treatment comprises 20 replicate runs with unique random seeds. Figures 5 and 6 plot the evolutionary trajectories of distance and efficiency, respectively. Individuals evolve viable gaits with the best individual per replicate traveling 23.5 units (7.83 body lengths) on average during a simulation. Units of distance are a simulation based metric and do not correspond to a physical dimension. For reference, the main body is 3 units long, 1 unit wide, and 0.5 units tall. Moreover, the movements increase in efficiency over evolutionary time, although this is not selected for. Instead, it arises as distance traveled increases, apparently as the leg movements evolve coordination.

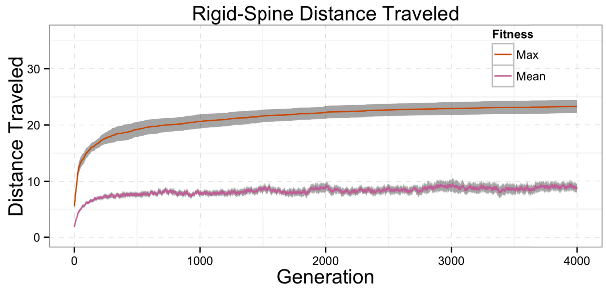


Figure 5: The mean maximum and average fitness across 20 replicate runs for the *Rigid-Spine* treatment. The shaded areas in this and subsequent plots represent the 95% confidence intervals.

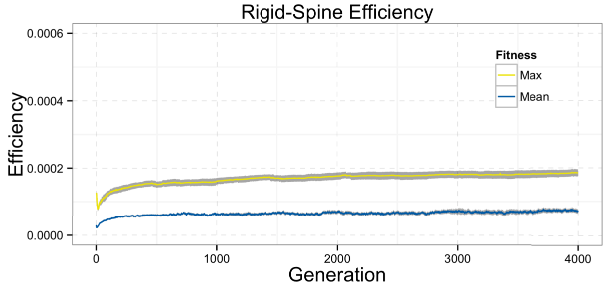


Figure 6: The mean maximum and average efficiency across 20 replicate runs for the *Rigid-Spine* treatment. Efficiency is measured as the distance traveled per unit of force exerted during a simulation. Efficiency rises slightly over evolutionary time although it is not selected for.

Flexible Spine In the *Flex-Spine* treatment, the torso is divided into three segments with a fixed spring coefficient between segments. The evolutionary trajectory of distance traveled and efficiency are plotted in Figures 7 and 8, respectively. When compared to the *Rigid-Spine* treatment, the flexible spine significantly improves both distance traveled and efficiency ($p < 0.001$ Mann-Whitney-Wilcoxon Test for both). The best individuals across replicates move 32.5 units (10.83 body lengths) during a simulation. The addition of a passively flexible spine appears to enable both higher performing and more efficient gaits, as efficiency is more than double that of a fixed spine animat.

Evolvable Flexibility In the final treatment, we evolve the level of flexibility in the spine along with the ANN controller. Both spine joints in an animat have the same evolved level of flexibility. The evolutionary trajectories of distance traveled and efficiency are plotted in Figures 9 and 10, respectively. This treatment produces the highest performing and most efficient individuals. The best individual per replicate averages 33.5 units traveled (11.16 body lengths).

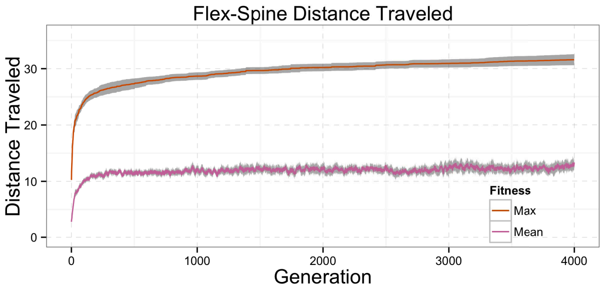


Figure 7: The mean maximum and average fitnesses across 20 replicate runs for the *Flex-Spine* treatment.

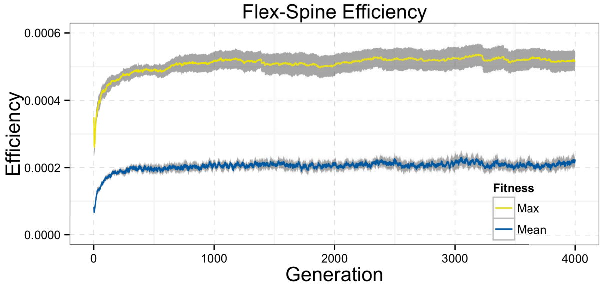


Figure 8: Mean maximum and average efficiency across 20 replicate runs for the *Flex-Spine* treatment.

Compared to the *Flex-Spine* treatment, the evolvable flexibility significantly improves performance ($p < 0.001$ Mann-Whitney-Wilcoxon Test) with no significant difference in efficiency ($p = 0.05589$). The ability to evolve flexibility is beneficial, but we observe that the individuals do not converge to a specific level of flexibility. Instead, high performing individuals evolve different levels of spine flexibility. Thus, it appears that the co-evolutionary process of control and morphology leads to more diversity in solutions compared to a specific level of flexibility.

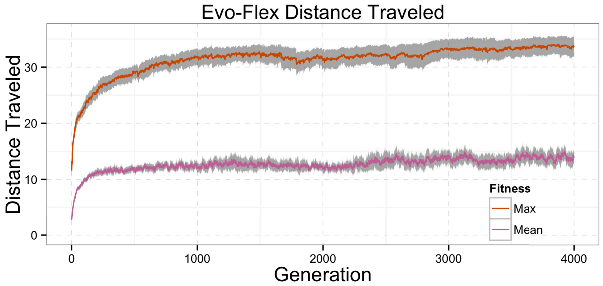


Figure 9: Mean maximum and average fitnesses across 20 replicate runs for the *Evo-Flex* treatment.

Figure 11 plots the performance of the farthest traveling individual from each of the 20 replicate runs in the *Evo-Flex* treatment. Evolved individuals fall into two categories of

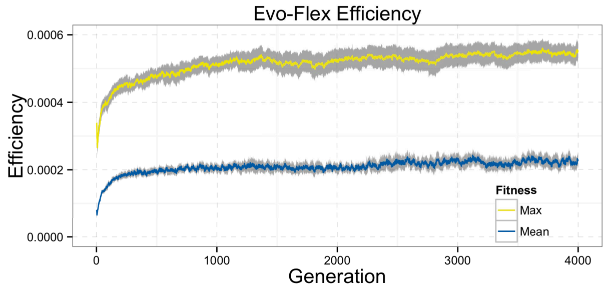


Figure 10: Mean maximum and average efficiency across 20 replicate runs for the *Evo-Flex* treatment.

spine flexibility. The highest performing individuals have relatively low spring coefficients (< 400), indicating highly flexible spines. Individuals in this category exhibit gaits with large deflection of the spine during movement, allowing the torso to act as a soft spring, folding and unfolding throughout the gait. The second set of individuals are in the midrange of flexibility (750 - 2000). In these individuals, spine flexibility is less pronounced, although still present. The spine acts more like a rigid spring, with only small deflections. Evolved gaits are similar to those of the individuals from the *Rigid-Spine* treatment. The shaded region, between spring coefficients of 400 and 750, in Figure 11 indicates an area where no replicate run's farthest traveling individual evolved. This area presumably prevents high performance in the evolved individuals. Additionally, we highlight this region in Figures 12 , 13, and 14. This area indicates a region in which no replicate run's farthest traveling individual evolved. This area presumably prevents high performance in the evolved individuals.

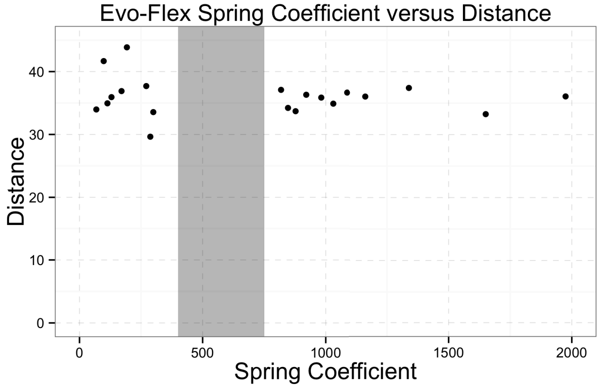


Figure 11: The distance traveled versus spine flexibility for the best individual from each replicate in the *Evo-Flex* treatment. The farthest traveling individuals have highly flexible spines as indicated by the low spring coefficient.

Figures 12 and 13, respectively, plot results for two replicates (12 and 4) of the farthest traveling individual per generation for two replicates from the *Evo-Flex* treatment. These two are representative of the remaining replicates. Figure 12 shows a replicate that evolves moderate spinal flexibility crossing from highly flexible individuals through the gap in flexibility. Furthermore, Figure 13 shows a more moderate progression through the shaded rectangle. Although this pattern shows that individuals can evolve inside this range of flexibility, the best individual for this replicate evolves a stiffer spine. The evolutionary trajectories of the replicates may explain the split between the highest performing individuals (Spring Coefficient < 400) and those with moderate flexibility (Spring Coefficient > 750) who are not as high performing. Apparently, it is difficult for individuals within this flexibility range to travel far.

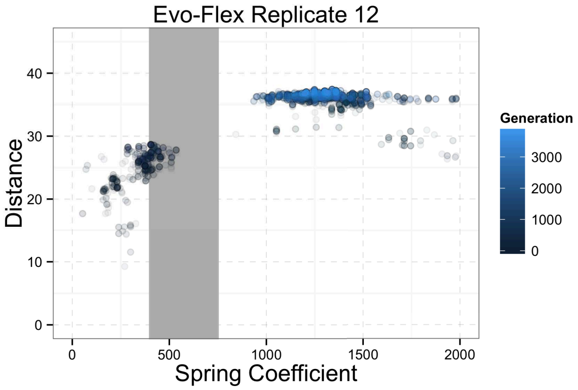


Figure 12: The farthest traveling individual per generation for a single replicate from the *Evo-Flex* treatment. Brighter colors represent individuals who evolve in later generations.

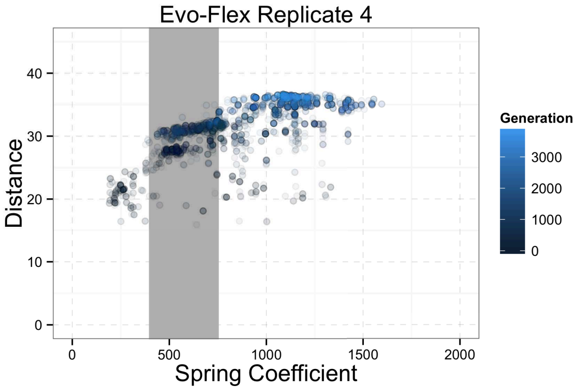


Figure 13: The farthest traveling individual per generation for a single replicate from the *Evo-Flex* treatment.

Figure 14 plots the efficiency versus evolved spring coefficient of the farthest traveling individuals from each replicate in the *Evo-Flex* treatment. Although not a target of the selection process, there appears to be an inherent efficiency in evolved solutions. In contrast to the distance traveled results, efficiency scores are not biased toward highly flexible spines, however. Instead, evolved individuals exhibit high energy efficiency across the range of spring coefficients. Furthermore, individuals with similar flexibility do not necessarily have similar performance suggesting that the controller also influences performance.

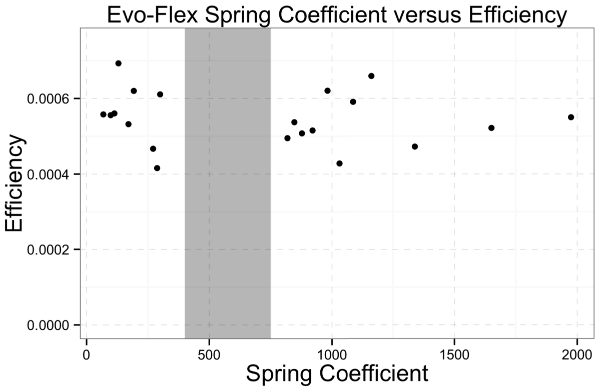


Figure 14: Efficiency of the farthest traveling individual from each replicate in the *Evo-Flex* treatment compared to their spine flexibility. In contrast to Figure 11, efficiency does not appear to depend on spine flexibility.

Performance and Efficiency Comparisons Figure 15 plots efficiency versus distance traveled of the farthest traveling individuals from each replicate in the three treatments. The *Evo-Flex* treatment produces the farthest traveling individuals. Boxplot distributions for distance traveled are shown in Figure 16. Applying a pairwise Mann-Whitney-Wilcoxon Test, finds all three pairings to be significantly different ($p < 0.001$). The addition of a flexible spine, even one with a predetermined flexibility, results in a significant improvement in performance.

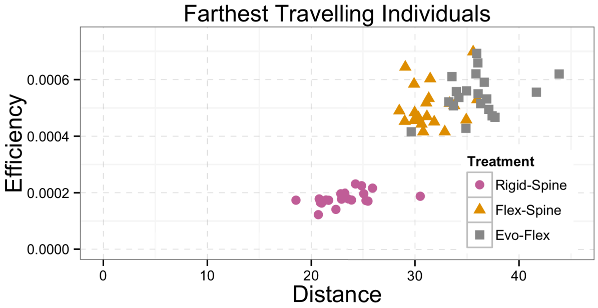


Figure 15: The farthest traveling individuals from each of the three treatments conducted in this study. Evolving the flexibility of the spine produces the highest performing, and most efficient individuals across all three treatments.

In addition to performance gains, flexible spines (*Flex-Spine*, *Evo-Flex*) also result in a significant increase in efficiency when compared to the *Rigid-Spine* treatment ($p <$ 0.001 Wilcoxon Test), see Figure 17. However, there is no significant difference in efficiency between the farthest traveling individuals from the *Flex-Spine* and *Evo-Flex* treatments ($p = 0.05589$). Still, evolving the flexibility of the spine results in the best individuals in terms of combined performance and efficiency.

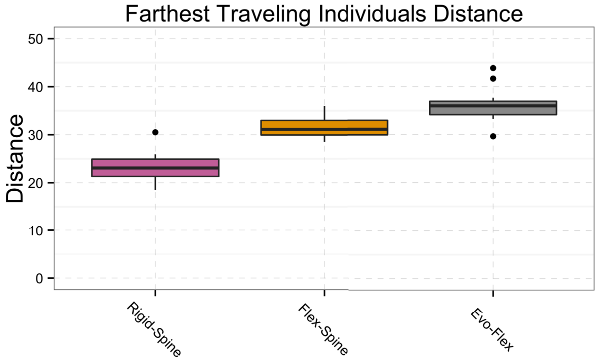


Figure 16: The performance distribution of the farthest traveling individual from each replicate across the three treatments.

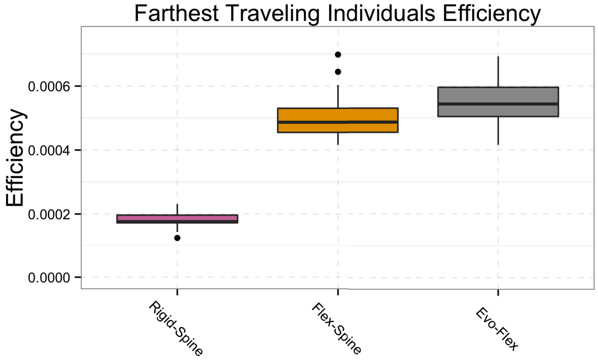


Figure 17: Distribution of efficiency in the farthest traveling individual from each replicate across the three treatments.

Body Position One way to compare gaits is to measure the height of the torso over the evaluation period. Figure 18 shows the mean torso height across replicates for each of the three treatments. The periodicity evident in the plot is due to the oscillating input signal provided to the evolved ANNs. Torso height for the flexible spine quadrupeds is measured as the position of the middle component of the torso, which corresponds to the measured position of the torso in the rigid spine animat. Two differences arise between flexible and rigid spines. First, the torso of individuals with flexible spines is generally higher than those with rigid spines. This suggests a posture with the legs under the robot as in walking or running, and less contact of the torso with the ground. Second, the amplitude of the rigid spine individuals is greater than those with flexible spines, indicating an increased vertical movement for those with rigid spines.

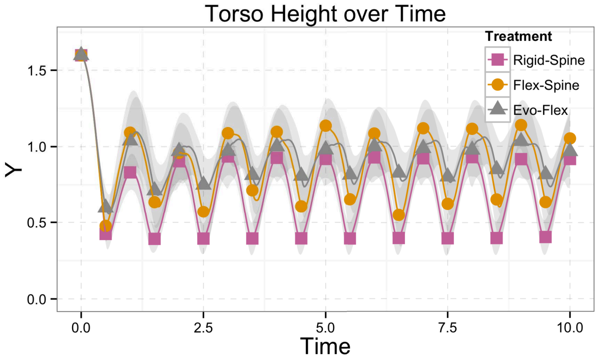


Figure 18: Mean torso height over simulation time across replicates for the three treatments. Error bars indicate the 95% confidence intervals.

Further analysis of the difference in amplitude is provided in Figure 19. Here, we calculate the mean position of the torsos per treatment normalizing the data to be the absolute deviation from the mean. This figure shows the vertical movement of the torsos over time. As shown, the *Rigid-Spine* and *Flex-Spine* treatments have similar displacements from their respective means. Indeed, a Mann-Whitney-Wilcoxon Test indicates that there is no significant difference in the amplitudes between these two treatments ($p = 0.06316$). However, the displacement of the *Evo-Flex* treatment is significantly lower than that of both *Rigid-Spine* and *Flex-Spine* treatments ($p < 0.001$ for both). In short, individuals with both evolved control and flexibility have the least amount of vertical movement in their torso.

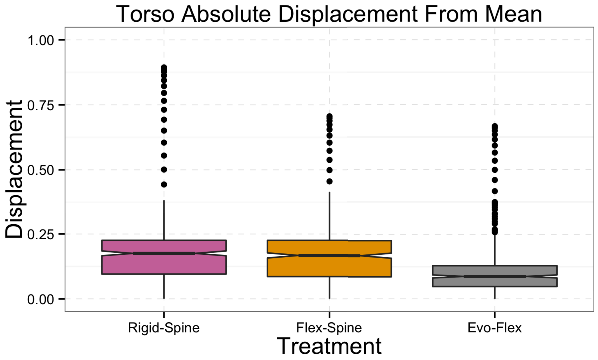


Figure 19: Mean normalized torso displacement for 20 replicate runs across the three treatments. The average torso height was determined per treatment and displacements recorded. The *Evo-Flex* individuals generally have less vertical movement throughout their gait.

This result is consistent with prior research showing that reducing the vertical movement of mass during locomotion can increase the energy efficiency of a legged robot (Ackerman and Seipel, 2013). In our study, the torso is the highest mass component of the quadruped, with the flexible spine acting to reduce its vertical movement. The result is greater distances traveled in a more energetically efficient gait. In biological organisms, minimizing collision force with the ground leads to efficient gaits (Bertram and Hasaneini, 2013). Collisions are typically dampened by minimizing the vertical displacement of the center of mass. After examining the body position over time, it appears that the evolved individuals with a flexible spine have gaits that minimize vertical displacement of the torso.

Conclusions

Biological organisms exhibit a level of agility and dexterity unparalleled in their robotic counterparts. In this paper, we have introduced a flexible spine to a quadruped animat. Results of evolutionary runs indicate that a flexible spine significantly increases performance in the evolved gaits. Furthermore, efficiency is also significantly improved, even without an explicit pressure to do so. It remains to be seen if additional benefits are to be gained by adding flexibility throughout the body, including legs and feet, which are rigid in this study.

In this preliminary investigation, we evolve individuals solely on a distance traveled metric. Efficiency is evaluated only after evolution. Future work will address incorporating efficiency as a metric for evolution, in addition to performance goals. Additionally, we plan to investigate passive components in other animats to help identify universal principles to apply to the robot design process.

Acknowledgments

The authors would like to thank Anthony J. Clark and gratefully acknowledge the BEACON Center for the Study of Evolution in Action, which directly supported this study and fosters an environment for interdisciplinary collaboration. This research also has been supported in part by U.S. National Science Foundation grants CNS-1059373, CNS-0915855, CNS-0751155, CCF-0820220, and DBI-0939454. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or other research sponsors.

References

Ackerman, J. and Seipel, J. (2013). Energy efficiency of legged robot locomotion with elastically suspended loads. *IEEE Transactions on Robotics*, 29(2):321–330.

Alexander, R. M. (1984). Elastic energy stores in running vertebrates. *American Zoologist*, 24(1):85–94.

Alexander, R. M. (1988). Why mammals gallop. *American Zoologist*, 28(1):237–245.

Anderson, J. M. and Chhabra, N. K. (2002). Maneuvering and stability performance of a robotic tuna. *Integrative and Comparative Biology*, 42(1):118–126.

Berns, K., Ilg, W., Deck, M., and Dillmann, R. (1998). The mammalian-like quadrupedal walking machine BISAM. In *Proceedings of the 1998 5th International Workshop on Advanced Motion Control*, pages 429–433, Coimbra, Portugal.

Bertram, J. E. A. and Hasaneini, S. J. (2013). Neglected losses and key costs: tracking the energetics of walking and running. *Journal of Experimental Biology*, 216(6):933–938.

Bongard, J. (2011). Morphological change in machines accelerates the evolution of robust behavior. *Proceedings of the National Academy of Sciences*, 108(4):1234–1239.

Cavagna, G. A., Heglund, N. C., and Taylor, R. C. (1977). Mechanical work in terrestrial locomotion: two basic mechanisms for minimizing energy expenditure. *American Journal of Physiology*, 233(5):R243–261.

Chaoui, H., Sicard, P., and Gueaieb, W. (2009). ANN-based adaptive control of robotic manipulators with friction and joint elasticity. *IEEE Transactions on Industrial Electronics*, 56(8):3174–3187.

Clark, A., Wang, J., Tan, X., and McKinley, P. (2014). Balancing performance and efficiency in a robotic fish with evolutionary multiobjective optimization. In *Proceedings of the 2014 IEEE International Conference on Evolvable Systems (ICES)*, pages 227–234, Orlando, Florida, USA.

Clark, A. J., Moore, J. M., Wang, J., Tan, X., and McKinley, P. K. (2012). Evolutionary design and experimental validation of a flexible caudal fin for robotic fish. In *Proceedings of the 13th International Conference on the Simulation and Synthesis of Living Systems*, pages 325–332, East Lansing, Michigan, USA.

Clune, J., Beckmann, B. E., Ofria, C., and Pennock, R. T. (2009). Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2764–2771, Trondheim, Norway.

Culha, U. and Saranli, U. (2011). Quadrupedal bounding with an actuated spinal joint. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1392–1397, Shanghai, China.

Doncieux, S. and Mouret, J. B. (2013). Behavioral diversity with multiple behavioral distances. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pages 1427–1434, Cancun, Mexico. IEEE.

Epstein, M., Colgate, J., and MacIver, M. (2006). Generating thrust with a biologically-inspired robotic ribbon fin. In *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2412 –2417, Beijing, China.

Geijtenbeek, T., van de Panne, M., and van der Stappen, A. F. (2013). Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics*, 32(6):1–11.

Hildebrand, M. (1959). Motions of the running cheetah and horse. *Journal of Mammalogy*, 40(4):481–495.

Ijspeert, A. J., Crespi, A., and Cabelguen, J.-M. (2005). Simulation and robotics studies of salamander locomotion. *Neuroinformatics*, 3(3):171–195.

Krishnamurthy, P., Khorrami, F., De Leeuw, J., Porter, M. E., Livingston, K., and Long, J. (2010). An electric ray inspired biomimetic autonomous underwater vehicle. In *Proceedings of the American Control Conference (ACC)*, pages 5224–5229, Baltimore, Maryland, USA.

Leeser, K. F. (1996). Locomotion experiments on a planar quadruped robot with articulated spine. Master's thesis, Dept. of Mechanical Engineering, Massachusetts Institute of Technology.

Lessin, D., Fussell, D., and Miikkulainen, R. (2013). Open-ended behavioral complexity for evolved virtual creatures. In *Proceedings of the 2013 ACM Genetic and Evolutionary Computing Conference*, pages 335–342, Amsterdam, Netherlands. ACM.

Moore, J. M. and McKinley, P. K. (2012). Evolving flexible joint morphologies. In *Proceedings of the 2012 ACM Genetic and Evolutionary Computing Conference*, pages 145–152, Philadelphia, Pennsylvania, USA. ACM.

Moore, J. M. and McKinley, P. K. (2013). Evolution of an amphibious robot with passive joints. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pages 1443–1450, Cancun, Mexico. IEEE.

Moore, J. M. and McKinley, P. K. (2015). Evolving energy-efficient locomotion in legged robots. Technical Report MSU-CSE-15-6, Computer Science and Engineering, Michigan State University, East Lansing, Michigan. submitted for publication.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence and Technology of Self-Organizing Machines*. The MIT Press.

Park, S. and Lee, Y.-J. (2007). Discontinuous zigzag gait planning of a quadruped walking robot with a waist-joint. *Advanced Robotics*, 21(1-2):143–164.

Paul, C. (2006). Morphological computation: A basis for the analysis of morphology and control requirements. *Robotics and Autonomous Systems*, 54(8):619 – 630.

Rieffel, J. A., Valero-Cuevas, F. J., and Lipson, H. (2010). Morphological communication: exploiting coupled dynamics in a complex mechanical structure to achieve locomotion. *Journal of The Royal Society Interface*, 7(45):613–621.

Ruina, A., Bertram, J. E. A., and Srinivasan, M. (2005). A collisional model of the energetic cost of support work qualitatively explains leg sequencing in walking and galloping, pseudo-elastic leg behavior in running and the walk-to-run transition. *Journal of Theoretical Biology*, 237(2):170–192.

Sfakiotakis, M., Kazakidi, A., Chatzidaki, A., Evdaimon, T., and Tsakiris, D. (2014). Multi-arm robotic swimming with octopus-inspired compliant web. In *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 302–308, Chicago, Illinois, USA.

Silva, M. F., Machado, J. A. T., and Lopes, A. M. (2005). Modelling and simulation of artificial locomotion systems. *Robotica*, 23:595–606.

Sims, K. (1994). Evolving 3D morphology and behavior by competition. *Artificial Life*, 1(4):353–372.

Smith, R. (2013). Open Dynamics Engine, http://www.ode.org/.

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127.

Valsalam, V. K. and Miikkulainen, R. (2008). Modular neuroevolution for multilegged locomotion. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, pages 265–272, Atlanta, GA, USA. ACM.

Van Ham, R., Sugar, T. G., Vanderborght, B., Hollander, K. W., and Lefeber, D. (2009). Compliant actuator designs. *IEEE Robotics & Automation Magazine*, pages 81–94.

Evolving Collective Behaviors With Diverse But Predictable Sensor States

Payam Zahadat[1], Heiko Hamann[2] and Thomas Schmickl[1]

[1]Artificial Life Lab of the, Department of Zoology, Karl-Franzens University Graz, Graz, Austria, payam.zahadat@uni-graz.at
[2] Department of Computer Science, University of Paderborn, Paderborn, Germany

Artificial evolution of collective behaviors can be implemented in different ways. Here, we extend the approach of Hamann (2014). The underlying concept is that the brain is permanently trying to predict future perceptions and a state of well-being is a state that allows for precise predictions (Friston, 2010). Striving for maximal prediction success needs to be complemented by a force that implements curiosity and exploration. In this abstract we present an extended method *diverse-prediction* that rewards not only for correct predictions but also for each visited sensory state. This proves to be a better approach compared to the method *prediction* (Hamann, 2014). As in the preliminary work (Hamann, 2014), we evolve pairs of artificial neural networks (ANN). One is predicting future sensor input and the other ANN outputs the next action. In our case study we simulate a homogeneous swarm (all agents share the same genome) of $N = 20$ agents (1st concept of population) that move in 1-d on a ring of circumference L. The agents have 4, discrete sensors covering 4 regions of the agent's vicinity and output 1 for 'there is at least 1 neighbor' or 0 otherwise. The available actions are: move forward or invert the heading. We evolve pairs of ANN with a population of size 50 (2nd concept of population) for 75 generations in 200 independent runs for each tested setting. Fitness for *prediction* rewards good predictions of sensor values. Fitness for *diverse-prediction* rewards for visiting more sensor states (combinations of sensor values) and making good predictions in those states. For that, we sum up the ratios between the number of correct predictions for each visited sensor state and the number of visits of that state over the whole swarm over time. Generally a desired result is a well explored behavior space combined with precise predictions. We check the degree of exploration by investigating a 2-d projection of behavior space. One dimension is covered distance which is the sum of the distances covered by the agents normalized by the considered time period and swarm size. Second dimension is the largest cluster size which is the maximum number of agents within sensor range normalized by swarm size. Fig. 1 (top) shows results for $L = 50$ but 3 settings were tested: high ($L = 5$), medium ($L = 20$), low agent density (L=50). We compare behavior distributions for *diverse-prediction*, *prediction*, and a control population of random ANNs. The projected behavior space is divided into a 5×15 grid and Pearson's χ^2 test is used to compare with a uniform distribution (UD). All 3 are far from UD ($P > 0.99$). See Fig. 1 (bottom) for the χ^2 values. We find that our new method *diverse-prediction* is best (smaller values mean the distribution is closer to UD which is better).

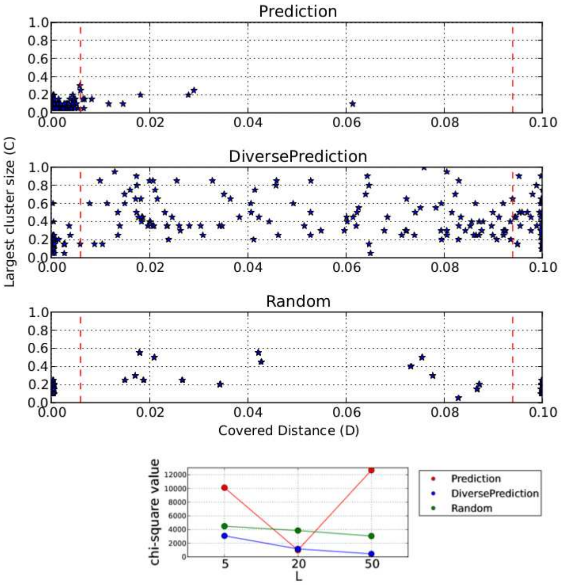


Figure 1: top: results of evolved behaviors for L=50; bottom: results from chi-squared test for $L \in \{5, 20, 50\}$.

Acknowledgments. Supported by: EU-H2020 project 'florarobotica', no. 640959; EU-H2020 project 'subCULTron', no. 640967; EU-ICT project 'ASSISI_bf', no. 601074; The authors would also like to thank Martin Stefanec for his helps.

References

Friston, K. (2010). The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138.

Hamann, H. (2014). Evolution of collective behaviors by minimizing surprise. In *14th Int. Conf. on the Synthesis and Simulation of Living Systems (ALIFE 2014)*, pages 344–351.

Learning by Stimulation Avoidance as a Primary Principle of Spiking Neural Networks Dynamics

Lana Sinapayen[1], Atsushi Masumori[1], Nathaniel Virgo[2], and Takashi Ikegami[1]

[1]The University of Tokyo, Ikegami Laboratory, Tokyo, Japan
[2]Earth-Life Science Institute, Tokyo Institute of Technology, Tokyo, Japan
lana@sacral.c.u-tokyo.ac.jp

Abstract

Practical implementation of the concept of reward has deep implications on what artificial-life based systems can learn and how they learn it. How can a system distinguish between useful behavior and harmful behavior? In this paper we implement reward/punishment as the removal/application of a stimulation to a recurrent spiking neural network with spike-timing dependent plasticity. This implementation embodies the concept of reward at the level of the neuron, making learning mechanisms ubiquitous to the network. We show that this low-level learning scales up to the network level: the network learns arbitrary spatio-temporal firing patterns purely by interacting with the environment, from a random initial state where virtually no knowledge is available. This approach yields fast, noise-robust results.

Introduction

Learning mechanisms discovered in biological neural networks are often translated into hypotheses that can be implemented and tested in artificial neural networks (Sejnowski et al., 1988). The dynamics of artificial neural networks have recently grown much closer to their biological counterpart, thanks to the introduction of biologically plausible models of spiking neurons (Izhikevich, 2004; Brette et al., 2007). These refined models could facilitate the exchange of concepts between artificial and biological networks. These exchanges lead to stronger theoretical understanding of the biological networks, and sometimes to real world applications in Artificial Intelligence (Kawato and Samejima, 2007; Cheng et al., 2007).

Rather than studying biological networks in vivo, networks of cortical neurons cultured ex vivo are often used to study biological learning mechanisms. Cultured networks are easier to study because they are composed of relatively few neurons, and are isolated from most external influences. Despite being much simpler than actual brains, they retain some important properties of in vivo neural networks (Canepari et al., 1997), among which: type and distribution of cells, high connectivity, spontaneous activity, diversity of firing patterns.

The research presented in this paper is based on the findings of Shahaf and Marom on ex vivo neural networks (Shahaf and Marom, 2001; Marom and Shahaf, 2002). They demonstrate that an ex vivo neural network can learn a desired behavior if stimulation removal is used as a reward, following this protocol: (1) Apply a low frequency, focal electrical stimulation to the network. (2) When the desired behavior appears, remove the stimulation. After several of these “training sessions”, the network learns to produce the desired behavior in response to the stimulation. In practice, the authors show that the network learns to produce spikes at a determined location and in a determined time window, in response to a stimulation applied at a different location in the network.

Despite the promising potential of the findings of Sahaf and Marom, it seems that the applications, limits and explanatory mechanisms of the method were not further studied in cultured or artificial neural networks. In Marom and Shahaf (2002), the authors cite two different classes of reward theory: “[M]apping the behavioral concept of reward to a neural entity that strengthens a subset of synapses based on past performance of the neural system” and “In contrast, the Stimulus Regulation Principle advocates that neural connectivity changes are due to the persistence of a driving stimulus [. . .]. If the output of the system changes the driving stimulus by its removal, [. . .] the system is ‘frozen’ in its last conformation; no specific cellular and synaptic reward mechanism needs be postulated.” The authors invoke the Stimulus Regulation Principle to explain their results, but the actual mechanisms that could be implementing this principle in the biological network are not explored.

In this paper, we implement, extend and give some empirical justification to a new principle termed “Learning by Stimulation Avoidance” (LSA) using a small artificial neural network. We define the LSA principle as the reinforcement of behaviors leading to a decrease of stimulation, and the weakening of behaviors leading to an increase in stimulation. Implemented at the level of the neurons, LSA has the following properties: learning can be obtained selectively at a random location of the network if the stimulation is focal; LSA does not require the existence of a specific neural “rewarding module”; both population coding and time coding can be obtained as a result of LSA.

We postulate that Hebbian learning rules (Hebb, 1949) are not only sufficient to implement LSA in a spiking neural network, but that the use of spiking neural networks actually leads to the following emergent properties of Hebbian rules: exploration of the space of possible connections, stimuli differentiation, active strengthening of synapses after the execution of desired behavior, and synaptic pruning after the execution of undesirable behavior. LSA therefore relies on principles from both classes of reward theory cited by Marom, with the important addition of synaptic pruning.

Hebbian rules are often inaccurately summarized with the sentence "cells that fire together, wire together." This sentence originates from the observation that in biological neural networks, when two neurons connected by a synapse fire at a short time interval from each other, their synaptic strength will change (Markram et al., 1997). But the sentence incorrectly pictures the actual interactions between two connected excitatory neurons:

- If neuron A fires just **before** neuron B, the weight of the synapse conducting signals from A to B will **increase**.
- If neuron A fires just **after** neuron B, the weight of the synapse conducting signals from A to B will **decrease**.

Therefore the actual dynamics of Hebbian rules depend on the delay between the two spikes and the timing of those spikes: cells that do fire together might well get unwired from each other. Furthermore, there are different mechanisms for inhibitory neurons leading to even more complex synaptic weight dynamics (Caporale and Dan, 2008). In this paper, we use Spike-Timing Dependent Plasticity (STDP, Caporale and Dan (2008); Song et al. (2000)) as an implementation of Hebbian rules for spiking networks. We focus on STDP between excitatory neurons exclusively. We demonstrate three hypotheses concerning LSA:

1. STDP alone is sufficient to realize LSA at the level of an individual synapse.
2. STDP-enabled neurons are able to react selectively to simultaneous simulations in several synapses.
3. Therefore, STDP-based LSA scales up to the level of an entire network.

Additionnaly, we show that this implementation is robust to noise, has a high success rate and is computationally fast.

Design

Spike-Timing Dependent Plasticity

We implement STDP as proposed by Bush et al. (2010). The equations and resulting weight variation are shown in Fig. 1, representing a situation where a neuron N_A and a neuron N_B fire within s ms of each other. The weight of the synapse transmitting signals from N_A to N_B varies as:

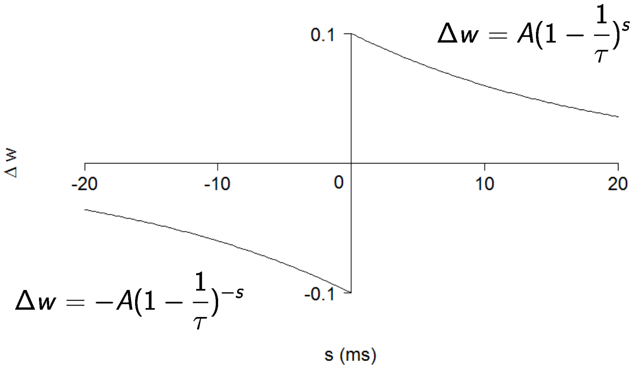


Figure 1: Spike-Timing Dependent Plasticity (STDP): weight variation Δw of the synapse from neuron A to neuron B depending on the relative spike timing $s = t_B - t_A$. $A = 0.1$; $\tau = 20$ ms.

$$w_t = w_{t-1} + \Delta w \ . \qquad (1)$$

We fix a maximum value to the weight: if $w > w_{max}$, w is reset to w_{max}. In the experiments identified as such, we also apply a decay function to all the weights in the network. The decay function is applied at each iteration t as:

$$\forall w_t, \ w_{t+1} = 0.9999995 w_t \ . \qquad (2)$$

In the experiments, we simulate either minimal networks of 2 or 3 excitatory neurons with one output synapse each, or fully connected networks of 100 neurons (Recurrent Neural Network, or RNN). The RNN is composed of 20 inhibitory neurons and 80 excitatory neurons. Synapses are modelized by directed connections between two neurons (self-connections are proscribed). Every neuron is connected to each other neuron by a synapse of synaptic weight w. The output weights are positive for excitatory neurons and negative for inhibitory neurons. The weights are initially set to random values (uniform distribution, $0 > w > 5$ for connections from excitatory neurons and $-5 < w < 0$ for connections from inhibitory neurons). In both the minimal network experiments and the RNN-based experiments, the weight of each connection between excitatory neurons is updated at each iteration of the simulation according to the STDP rule. We do not apply STDP on input and output connections of inhibitory neurons.

Network model

For the implementation of the network, we use the spiking model of cortical neuron proposed by Izhikevich (Izhikevich, 2003). This model has two main advantages: it can be tuned to accurately reproduce the dynamics of different types of cortical neurons, and it is computationally ef-

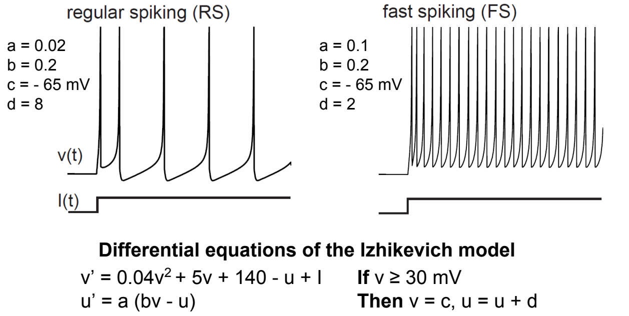


Figure 2: Equations and dynamics of regular spiking and fast spiking neurons simulated with the Izhikevich model. The spiking figures are reproduced with permission from www.izhikevich.com. (Electronic version of the figure and reproduction permissions are freely available at www.izhikevich.com)

ficient which allows for real time experimentation. It is currently the model with the dynamics the closest to biological neurons: networks based on this model exhibit biologically plausible behaviors, either spontaneously (delta and gamma rhythms, polychronization (Izhikevich, 2006)) or by relying on learning schemes (conditioning with delayed reward, (Izhikevich, 2007)).

The Izhikevich model has two main equations and a reset condition; the parameters and dynamics of regular spiking excitatory neurons and fast spiking inhibitory neurons are summarized in Fig. 2. We do not discuss here the details of the model; for the understanding of this paper it is sufficient to know that v (mV) is the membrane potential of the neuron, u the membrane recovery variable, I (mV) the sum of synaptic inputs and externally injected currents, and Δt (ms) is the time resolution. We fix the time resolution to $\Delta t = 1$ ms.

A specificity of the Izhikevich model is the role of "noisy thalamic inputs" (refered simply as "noise" in this paper). In this model, a noise variable m_i is added to the input of each neuron at each iteration. This noise regulates the global level of activity of the network: with no noise at all, the connections in a small network do not provide sufficent stimulation to lead to sustained activity. With too much noise, the network enters a state where all neurons are continuously spiking. We choose m as a variable with a zero-mean Gaussian distribution and a standard deviation of σ. m takes a different value at each iteration and for each neuron N_i.

When a neuron N_j fires, each other neuron has its input variable augmented by the weight of the input connection with N_j. Therefore at each iteration, the input I_i^* received by a neuron N_i from all other neurons is

$$I_i^* = \sum_{j=0}^{n} w_{j,i} \times f_j \ , \tag{3}$$

$$f_j = \begin{cases} 1, & \text{if } N_j \text{ is firing} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

with n the number of neurons in the network, $w_{j,i}$ the weight of the connection from N_j to N_i, and $w_{i,i} = 0$.

An external input e, for example corresponding to a stimulation from an electrode, can be added to the input of a neuron. The total input in one neuron at each iteration is therefore

$$I_i = I_i^* + e_i + m_i \ . \tag{5}$$

Experimental protocol

We perform two types of experiments. In the first type of experiment (Experiments 1 and 2), we study the basis of STDP-based LSA in a minimal network of 2 or 3 neurons. In the second type of experiment (Experiment 3), we show that LSA scales up to the level of a RNN with 100 neurons. The task that must be learned in the RNN experiments is synchronous population coding: we choose groups of neurons in the network and a firing pattern for each group. The combination of groups and firing patterns constitutes the "desired behavior" that the network must learn to exhibit in response to external stimulation.

The training protocol is as follows: (1) The experimenter defines the desired behavior; (2) External stimulation is applied until the desired behavior appears, or until a time delay is reached; (3) The stimulation is removed; (4) Step 2 and 3 constitute one training cycle; they are repeated until the delay between the beginning of the stimulation and the apparition of the desired behavior is systematically under a fixed value (behavior learned).

Experiments and Results

1. SDTP-based LSA in a single synapse

We design the first experiment to study the weight variation in one synapse between two neurons (Fig. 3). We control the external stimulation e_0 in N_0 under 3 conditions summarized in Fig. 4: (a) Stop the stimulation if N_1 fires; (b) Start the stimulation if N_1 fires; (c) Stimulate N_0 whatever the behavior of N_1. The delay between two training cycles is 30 ms (except in (c) where the stimulation goes uninterrupted). We set the stimulation as $e_0 = 2$ mV and the noise standard deviation as $\sigma = 10$. These parameters are chosen rather arbitrarily as there is no network effect to take into consideration. The initial weight is $w_{0,1} = 5$.

At the beginning of the experiment, the synaptic weight is comparatively low and N_1 fires in reaction to both the firing of N_0 and the high random noise m_1. STDP is applied to the

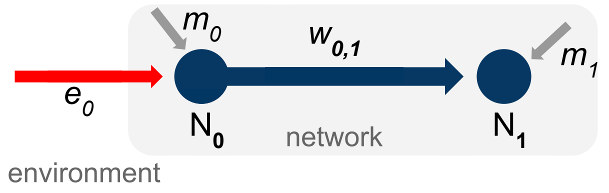


Figure 3: Experimental setup: the minimal network counts 2 neurons and 1 synapse. Random noise m is added as input to both neurons. An external stimulation e_0 is applied to N_0. The dynamics of e_0 depend on the experimental conditions.

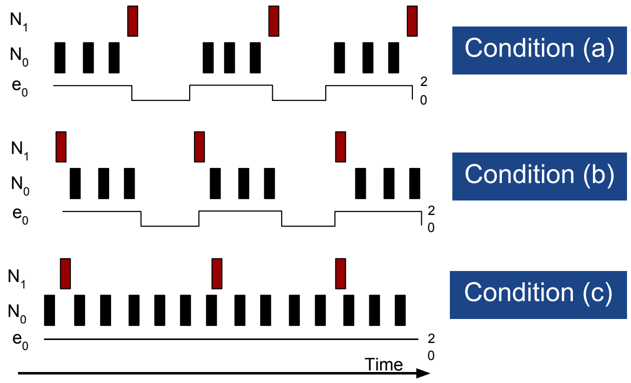


Figure 4: The three conditions used in Experiment 1 summarized as a raster plot. The rectangles represent neuronal spikes. The external stimulation e_0 causes N_0 to fire rhythmically; firing of N_0 causes stimulation in N_1. In real conditions, the noise and synaptic weight variations cause less regular spiking.

minimal network, causing the weight $w_{0,1}$ to vary according to the results in Fig. 5.

During a training cycle in (a), firing of N_1 causes the stimulation in N_0 to stop, therefore N_0 stops firing. N_0 lastly fired just before N_1 did, so as a result of STDP, $w_{0,1}$ is strengthened. This stronger weight in the synapse eventually leads to N_1 firing directly in reaction to the firing of N_0; the delay between the application of the stimulation to N_0 and the firing of N_1 decreases with time. In other words, the minimal network learns to immediately produce the behavior leading to stimulation removal. The learning consists in associating a given stimulus (external stimulation of N_0) with a behavior (firing of N_1).

In (b), random firing of N_1 causes the stimulation in N_0 to start (therefore N_0 starts firing). N_1 fired just before N_0 starts firing, so $w_{0,1}$ is decreased by STDP: the pre-synaptic neuron N_0 will have less and less influence on the post-synaptic neuron N_1. The synaptic weight finally reaches 0. The network learns to avoid the behavior that causes stimulation (firing of N_1). The same behavior that was learned in (a) is now avoided.

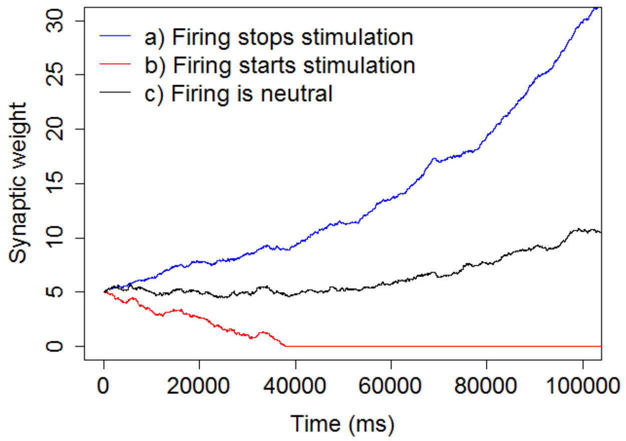


Figure 5: Weight variation in one synapse depending on the effect of post-synaptic neuron firing. The principle of LSA is verified: behaviors conducting to stimulation avoidance are reinforced via synaptic strengthening or synaptic pruning. The default dynamics of STDP lead to weight strengthening in neutral conditions, behavior which can be partly avoided by applying a decay function.

In (c), N_0 is continuously stimulated. The synaptic weight increases slowly but continuously: as long as the firing of N_1 is not clearly the cause of the external stimulation of N_0, the connexion will be slowly strengthened. The slow increase of the weight, as opposed to stable variations around the initial value of 5, is explained by two factors. First, N_0 contributes to the stimulation in N_1. Therefore, N_1 if more likely to fire after N_0 fired: spikes of N_1 will on average be closer to the last spike N_0 than to its next spike. This leads to $w_{0,1}$ being reinforced; in return, this stronger weight causes smaller time delays between the spikes of the two neurons. This can potentially be exploited as an exploratory behavior, but in practice, in large networks it leads to a state of weight saturation where all neurons are constantly firing. One way to deal with the issue is to apply a decay function on the weights. In our network, noise is mainly responsible for the exploration process, so we apply the decay function to avoid weight saturation.

This simple experiment with a minimal network validates Hypothesis 1: STDP alone is sufficient to realize LSA at the level of an individual synapse. In a minimal network with a single synapse, the synapse is strengthened to reinforce post-synaptic firing if it leads to removal of pre-synaptic stimulation; the same synapse is pruned if post-synaptic firing causes pre-synaptic stimulation. Therefore STDP is sufficient to realize Learning by Stimulation Avoidance at the level of a single synapse. Additionally, by running experiments with different values of external stimulation e_0 and

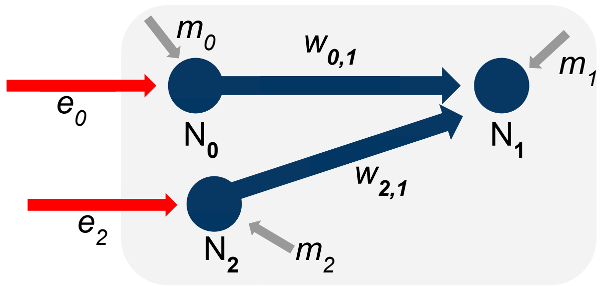


Figure 6: Augmented experimental setup: 3 neurons and 2 synapses. Random noise m is added as input to all neurons. The dynamics of the external stimulations e_0 and e_2 are different and depend on the experimental conditions.

noise m, we find that the learning speed tends to decrease when the noise level or the stimulation level are decreased. For example, reducing the noise standard deviation to $\sigma = 5$ and the stimulation to $e_0 = 1$ leads to a weight of only 10 in the reinforced synapse after 100 000 ms, compared to a weight of 30 in Experiment 1. Both high noise and high stimulation values tends to increase the firing rate of N_1, which increases the learning speed. This leads to the paradoxical observation that noise increases the performance of the minimal network. Furthermore, the results still hold for extremely low signal to noise ratio (high noise, low stimulation). In the next experiment, we validate Hypothesis 2 and show that the minimal network is capable of selective learning.

2. Effect-based differentiation of stimuli

In Experiment 2, we add one neuron to the minimal network (Fig. 6). The noise standard deviation is reduced to $\sigma = 5$ to account for the increased stimulation in the network (due to both $w_{2,1}$ and e_2). The external stimulations e_0 and e_2 vary independently between 0 mV and 2 mV. N_0 is stimulated until N_1 fires, then e_0 is stopped for 30 ms. N_2 is stimulated by e_2 during those 30 ms. So the firing of N_1 causes external stimulation in N_2, but stops external stimulation in N_0. The network must tell apart these influences despite the noise: we expect $w_{0,1}$ to increase as a realization of LSA, since firing of N_1 is beneficial to N_0 (it stops the external stimulation). Meanwhile, the firing of N_1 is detrimental to N_2, as it causes external stimulation: if LSA is realized, $w_{2,1}$ should decrease. These are indeed the results of the experiment, as shown in Fig. 7: $w_{0,1}$ (in blue) increases and $w_{2,1}$ (in red) decreases at the same time. This result is explained by the fact that despite the noise, there are overall more spikes of N_0 just before spikes of N_1 than just after, leading through STDP to an increase in weight. Similarly, there are overall more spikes of N_2 just after spikes of N_1 than just before, leading to an decreasing weight.

We also perform a variant of this experiment where the stimulation in N_0 stops 5 ms after the firing of N_1 (instead of stopping instantly). Therefore not only the causality between the spikes of N_1 and the end of the stimulation is delayed, but additionally the stimulations in the two presynaptic neurons N_0 and N_2 overlap for 5 ms. Despite these additional difficulties, the results stay qualitatively the same as in the original experiment, with an increased learning speed ($w_{2,1}$ reaches 0 at $t \approx 40\,000$ ms). The increase in speed is due to the increased firing rate of N_0 as a consequence of stimulation building up during the additional 5 ms.

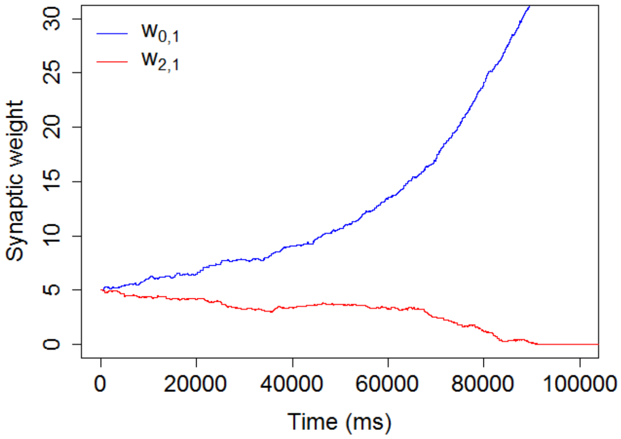


Figure 7: Parallel processing of two input synapses in one neuron.

These results validate Hypothesis 2: one neuron can receive simultaneous signals from two synapses and proceed to prune one while strengthening the other. Therefore the neuron will react differently to two stimulations with conflicting effects.

3. Synchronous population coding in a larger network

What works for one neuron may not work in a more complex network where all neurons influence each other, and where these influences are much more difficult to tell apart. In Experiment 3, we test the scalability of LSA in a network of 100 neurons, all connected to each other by synapses with initially random weights. Related works suggest that in large random networks, having around 20% of inhibitory neurons (as in the mammalian cortex) is important for the dynamics of the network (Izhikevich, 2003; Connors and Gutnick, 1990). Therefore we set 20 of the neurons (N_0 to N_{19}) as inhibitory neurons.

The experimental protocol is as follows: 10 neurons (N_{20} to N_{29}) are chosen as input neurons and externally stimulated ($e = 0.8$ mV). We monitor the activity in 20 output

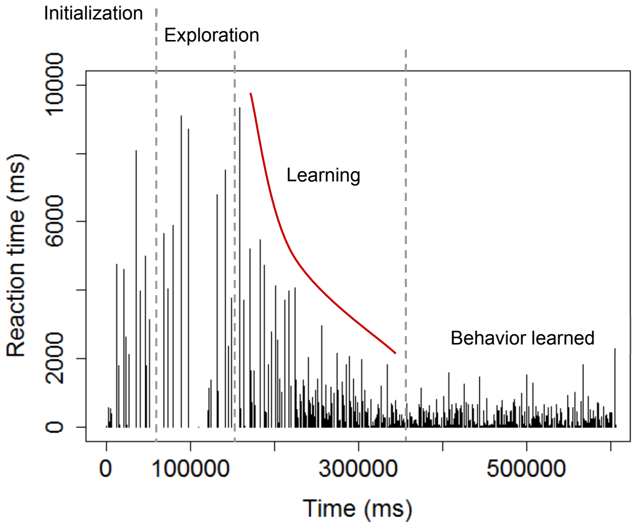


Figure 8: Time delay between the beginning of the stimulation and the apparition of the desired firing pattern. The network goes through different phases; a learning curve is clearly visible. By the end of the experiment, the network exhibits the desired firing pattern consistently and rapidly after the beginning of the stimulation.

neurons and define the task to be learned as follows: in a group of 10 output neurons (group A, N_{30} to N_{39}), at least 4 neurons should fire simultaneously at time t (resolution = 1 ms). In a different group of 10 neurons (group B, N_{40} to N_{49}) and at the same time t, less than 4 neurons should be firing. The condition on group B ensures that we avoid a trivial solution where all 100 neurons synchronize their spikes. When the desired firing pattern is obtained from the outputs, we stop stimulating the input neurons. If the desired firing pattern is not obtained after 10 000 ms, the stimulation is also stopped. In both cases, the stimulation starts again 500 ms after stopping. The experiment lasts 600 000 ms. The noise standard deviation is $\sigma = 3$ and the maximum weight is $w_{max} = 15$.

Fig. 8 shows the learning curve of the network. At the beginning of the experiment, the desired output is obtained at random delays after starting the stimulation. This is an initialization phase where Izhikevich networks are subject to a few highly synchronized bursts (see also Fig. 9). After the initialization phase, there is an exploration phase where the stimulation is often stopped because the maximum stimulation time is reached, and seldom because the desired output was recorded. The exploration phase is followed by a learning phase, where the learning curve decreases steeply to short reaction times. Towards the end of the experiment, the desired output is obtained within short reaction times (less than 2 000 ms after the start of the stimulation). Fig. 9 shows the evolution of the network's firing patterns. The initialization phase (a) is dominated by long, sparse, highly synchronized bursts involving all neurons. At the learning phase (b), these completely synchronized bursts have been replaced by more temporally distributed firing. After the desired behavior is learned in (c), the raster plot shows high global activity of the network, with short, strongly structured bursts.

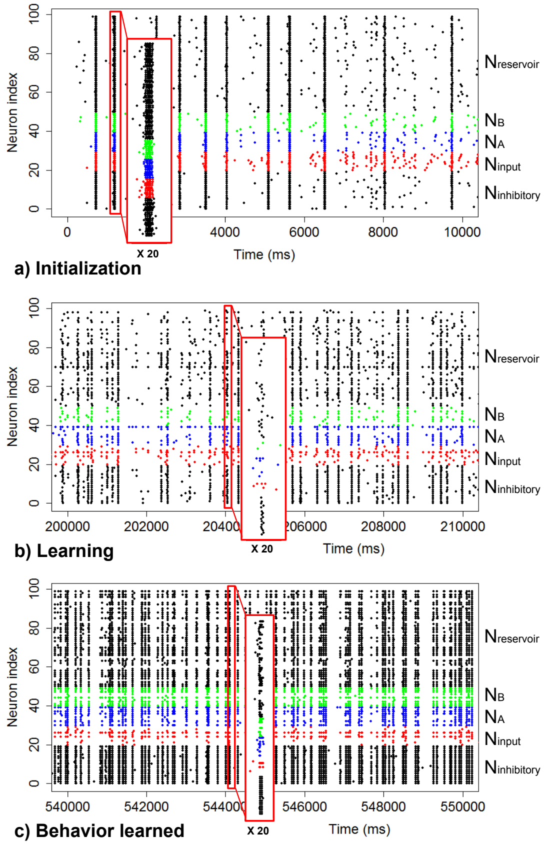


Figure 9: Raster plot showing the temporal firing of the network at different phases of the experiment. The global activity of the network shows a particularly clear distinction between the dynamics of the input neurons and the other groups (input neurons fire less often than other neurons, due to reduced input weights from the rest of the network). The insets show the gradual partial desynchronization of the different groups.

We ran the same experiment on 10 additional randomly generated networks (10 min of simulation time). Out of these 10 networks, 7 reached a reaction time systematically inferior to 2 000 ms after only 5 min of simulation. For these networks, the average reaction time after learning ($t > 5$ min) was 108 ms. The average reaction time of the remaining 3 networks was 1 056 ms after 5 min. All 10 networks showed an increase in the frequency of apparition of the desired output pattern and a decrease of the reaction time. To summarize, all networks successfully learned to exhibit the correct output pattern, each with an average reac-

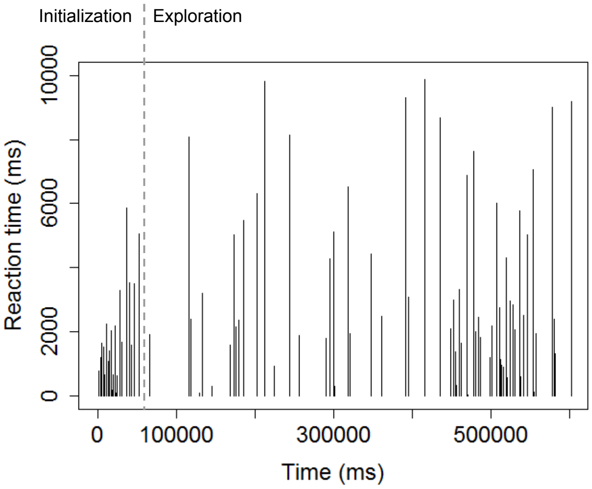


Figure 10: Results of Experiment 3 with no inhibitory neurons in the network. The initialization phase is as long as in the original conditions, but the network is stuck in the exploration phase and never reaches the learning phase.

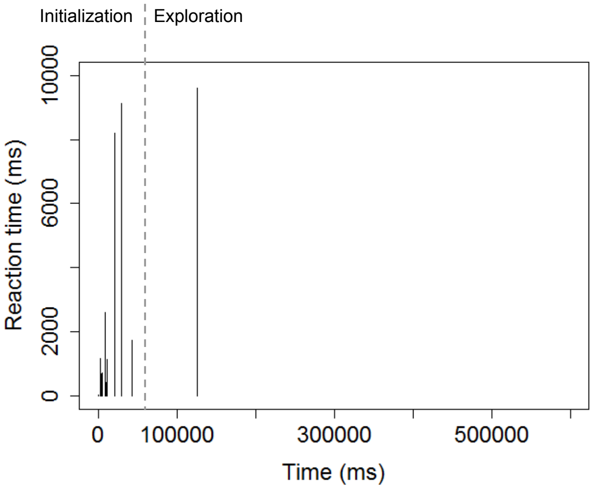


Figure 11: Results of Experiment 3 with $e = 0$ mV. The initialization phase is as long as in the original conditions, but with no way to differentiate the correct firing pattern from other firing patterns, the network has no proper exploration phase and does not reach the learning phase. The desired firing pattern stops being exhibited.

tion time inferior to 2 s. Therefore Hypothesis 3 is validated: LSA is scalable to networks of 100 neurons.

We performed different versions of this experiment, and found several interesting properties. First, the network never exhibits the trivial solution (all neurons firing simultaneously) even when there is no condition on group B. Secondly, the synaptic weights never follow a trivial distribution, where the input neurons directly cause the firing of the output neurons ($w_{input,A} = w_{max}$). The fact that the initial weight matrix is random is certainly the biggest factor explaining the complex distribution of the final weights. Thirdly, removing inhibitory neurons and setting all 100 neurons as excitatory prevents the network from learning (Fig. 10); all synaptic weights increase until saturation of the network. It is possible that inhibitory neurons prevent the network from forming too many recurrent excitation loops. Finally, we performed the experiment with no stimulation ($e = 0$ mV) as a null hypothesis. Fig. 11 shows that this condition does not lead to learning of the firing pattern, as expected.

Discussion

Although there is no consensus on the definition of learning, we believe that Experiments 1 and 2 qualify as learning in it simplest expression: the minimal network's behavior changes to adapt to external influences. One neuron learned to fire when the network was "rewarded" by stimulus removal, and the same neuron learned not to fire when the network was "punished" by the the application of a stimulus. These experiments serve as simplifications to explain the results of Experiment 3. In Experiment 3, the network starts in a random state, with no knowledge about itself, the environment, or the task that should be learned. The input neurons are no different from the two types of output neurons or from the reservoir neurons; all these groups are randomly chosen by the experimenter. The rule governing the stimulation pattern is also unknown by the network. Furthermore, initially most spikes in the network are due to strong random noise (the noise's standard deviation is more than three times as big as the stimulation). Noise is usually an issue in artificial networks, and in artificial systems in general. But here as in biological systems, the action of noise is mostly non-detrimental. It is even beneficial, as random firing is necessary to kick-start the learning process.

In true reinforcement learning fashion, the network learns entirely by interacting with the experimental environment and finds the correct output leading to stimulation avoidance. This "correct" output can be any combination of simultaneously firing/not firing neurons; the location and even the number of neurons implicated in the rule are unknown to the network. Despite the huge search space of all these possible combinations, and despite the impossibility of learning progressively as would be the case with a fitness function (in our case the output is either correct or incorrect, nothing in-between), the network correctly learns its ascribed task. Furthermore, the task is learned after less than 5 min of simulation time on average (about 3 min in real time using a rather slow laptop PC and a non-optimized programming language).

To summarize, our method allows temporal and population coding; it is robust to noise; it is simpler that most existing learning approaches for spiking networks and requires a smaller number of neurons; it is fast; it works with a wide range of parameters. In experiments yet unpublished, we find that the network can learn complex sequences of randomly chosen temporal patterns (polychronization) and can perform cost analysis over several stimulation sources. It also exhibits more classical forms of learning like Pavlovian conditioning.

On the other hand, one issue of this implementation is that it is not easily scalable. At the moment, either the noise level or the number of connections must be manually adjusted when working with networks of hundreds or thousands of neurons. This suggests that the network should be used in small modules or pathways loosely connected to each other. The true impact of all parameters (noise, stimulation, number of neurons and of connections, maximum weight) must be further studied in future works. Another issue not treated in this paper is the possibility to use other types of neurons with different dynamics.

Conclusion

The principle of Learning by Stimulation Avoidance yields promising results, and can be easily implemented in spiking networks. STDP might not be the only possible implementation of LSA, but associated to the Izhikevich network model, it leads to unique robustness and versatility. We believe that the implementation of LSA presented in this paper deserves to be submitted to more challenging tasks.

Acknowledgments

This work was supported by Grant-in-Aid for Scientific Research (Studies on Homeo-Dynamics with Cultivated Neural Circuits and Embodied Artificial Neural Networks; 24300080).

References

Brette, R., Rudolph, M., Carnevale, T., Hines, M., Beeman, D., Bower, J. M., Diesmann, M., Morrison, A., Goodman, P. H., Harris Jr, F. C., M., Z., Natschläger, T., Pecevski, D., Ermentrout, B., Djurfeldt, M., Lansner, A., Roche, l. A., Vieville, T., Muller, E., Davison, A. P., El Boustani, S., and Destexhe, A. (2007). Simulation of networks of spiking neurons: a review of tools and strategies. *Journal of computational neuroscience*, 23(3):349–398.

Bush, D., Philippides, A., Husbands, P., and O'Shea, M. (2010). Reconciling the STDP and BCM models of synaptic plasticity in a spiking recurrent neural network. *Neural computation*, 22(8):2059–2085.

Canepari, M., Bove, M., Maeda, E., Cappello, M., and Kawana, A. (1997). Experimental analysis of neuronal dynamics in cultured cortical networks and transitions between different patterns of activity. *Biological cybernetics*, 77(2):153–162.

Caporale, N. and Dan, Y. (2008). Spike timing-dependent plasticity: a Hebbian learning rule. *Annu. Rev. Neurosci.*, 31:25–46.

Cheng, G., Hyon, S.-H., Morimoto, J., Ude, A., Hale, J. G., Colvin, G., Scroggin, W., and Jacobsen, S. C. (2007). CB: A humanoid research platform for exploring neuroscience. *Advanced Robotics*, 21(10):1097–1114.

Connors, B. W. and Gutnick, M. J. (1990). Intrinsic firing patterns of diverse neocortical neurons. *Trends in neurosciences*, 13(3):99–104.

Hebb, D. O. (1949). *The organization of behavior*. Wiley, New York.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on neural networks*, 14(6):1569–1572.

Izhikevich, E. M. (2004). Which model to use for cortical spiking neurons? *IEEE transactions on neural networks*, 15(5):1063–1070.

Izhikevich, E. M. (2006). Polychronization: computation with spikes. *Neural computation*, 18(2):245–282.

Izhikevich, E. M. (2007). Solving the distal reward problem through linkage of stdp and dopamine signaling. *Cerebral cortex*, 17(10):2443–2452.

Kawato, M. and Samejima, K. (2007). Efficient reinforcement learning: computational theories, neuroscience and robotics. *Current opinion in neurobiology*, 17(2):205–212.

Markram, H., Lübke, J., Frotscher, M., and Sakmann, B. (1997). Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs. *Science*, 275(5297):213–215.

Marom, S. and Shahaf, G. (2002). Development, learning and memory in large random networks of cortical neurons: lessons beyond anatomy. *Quarterly reviews of biophysics*, 35(01):63–87.

Sejnowski, T. J., Koch, C., and Churchland, P. S. (1988). Computational neuroscience. *Science*, 241(4871):1299–1306.

Shahaf, G. and Marom, S. (2001). Learning in networks of cortical neurons. *The Journal of Neuroscience*, 21(22):8782–8788.

Song, S., Miller, K. D., and Abbott, L. F. (2000). Competitive Hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience*, 3(9):919–926.

Contagion on Networks with Self-Organised Community Structure

Alberto Antonioni[1], Seth Bullock[2], Christian Darabos[3], Mario Giacobini[4]
Bryan N. Iotti[4], Jason H. Moore[5], Marco Tomassini[1]

[1]Information Systems Department, Faculty of Business and Economics, University of Lausanne, Switzerland
[2]Institute for Complex Systems Simulation, University of Southampton, United Kingdom
[3] Computational Genetics Laboratory, Geisel School of Medicine, Dartmouth College, USA
[4]Computational Epidemiology Group, Department of Veterinary Sciences, University of Torino, Italy
[5]Institute for Biomedical Informatics, Perelman School of Medicine, University of Pennsylvania, USA

Abstract

Living systems are organised in space. This imposes constraints on both their structural form and, consequently, their dynamics. While artificial life research has demonstrated that embedding an adaptive system in space tends to have a significant impact on its behaviour, we do not yet have a full account of the relevance of spatiality to living self-organisation.

Here, we extend the REDS model of spatial networks with self-organised community structure to include the "small world" effect. We demonstrate that REDS networks can become small worlds with the introduction of a small amount of random rewiring. We then explore how this rewiring influences two simple dynamic processes representing the contagious spread of infection or information.

We show that epidemic outbreaks arise more easily and spread faster on REDS networks compared to standard random geometric graphs (RGGs). Outbreaks spread even faster on small world REDS networks (due to their shorter path lengths) but initially find it more difficult to establish themselves (due to their reduced community structure). Overall, we find that small world REDS networks, with their combination of short characteristic path length, positive assortativity, strong community structure and high clustering, are more susceptible to a range of contagion dynamics than RGGs, and that they offer a useful abstract model for studying dynamics on spatially organised living systems.

Introduction

The structure of living systems such as cells, brains, organisms, colonies and ecosystems is neither random, where all interactions between system components are equally likely, nor regular, where interactions between system elements are arranged uniformly. Rather, natural systems tend to exhibit complex structure, featuring clustering and modularity.

Despite this, many models of living systems either assume that the system is well-mixed, with every interaction between system components equally valid, or specify that the interactions within a system are embedded on a regular lattice. The behaviour of such models is sensitive to these choices regarding how interactions are structured, with, for instance, spatial embedding often encouraging complex organised behaviour where it would not otherwise arise (e.g., Boerlijst and Hogeweg, 1991; Di Paolo, 2000; Buckley et al., 2010).

To some extent, the complex structure of living systems arises as a consequence of adaptation to specific circumstances, but to some extent it is present as a consequence of more generic processes of self-organisation operating under physical constraints (Bullock and Buckley, 2009; Bullock et al., 2010; Bartlett and Bullock, 2014). For example, the fact that living systems are embedded in space impacts on the kind of forms that they may readily adopt: not every component of a spatial system may interact directly with every other component; some components are necessarily central, while others must be peripheral, etc.

In this paper we are interested in the impact that these types of constraints have on system structure, and the resultant consequences for system dynamics. We pursue this idea in the context of REDS[1], a spatially embedded network model that readily exhibits some of the key features of social networks: high clustering, right-skewed degree distribution, positive degree assortativity, and strong community structure (Antonioni et al., 2014).

We first introduce the REDS model and explore how it is affected by the introduction of random rewiring. We demonstrate that rewired REDS networks can be small worlds, i.e., that they exhibit the combination of strong clustering and short characteristic path length that is typical of some social and biological systems. Subsequently, we explore the influence of REDS network structure on the dynamics of two simple contagion processes, one representing the spread of an infectious disease and one representing the invasion of a

[1] The REDS acronym relates to the four aspects of a social system that influence network topology within the: the social *Reach*, *Energy*, and *Synergy* parameters of the network, and the *Distance* between pairs of nodes.

beneficical mutation. We are able to demonstrate that the combination of strong community structure and short path lengths in small world REDS networks facilitates the spread of contagion. The paper closes with a discussion of these results.

Previous Work

While work on relational (non-spatial) networks dominates network science, there is growing interest in spatially constrained networks (see Barthélemy, 2011, for a recent review of the field). In these models nodes are located in some metric space and connections between nodes are in some way influenced by the distance separating them (see Boguñá et al., 2004; Wong et al., 2006; Serrano et al., 2008; zu Erbach-Schoenberg et al., 2014, for some recent examples). REDS networks fall within this category and, like many spatial networks, are an extension of Random Geometric Graphs (RGGs), the canonical spatial network model.

A standard RGG can be constructed by distributing N points uniformly at random in some topological space, e.g., the two dimensional unit square, and connecting all pairs of nodes that are separated by a Euclidian distance less than a fixed threshold, R. There is an extensive literature on random geometric graphs, particularly in the context of continuum percolation (Dall and Christensen, 2002; Penrose, 2003; Barthélemy, 2011). The degree distribution of a RGG is Poisson with mean equal to $N\pi R^2$ (Dall and Christensen, 2002). The clustering coefficient of a RGG tends to $1 - \frac{3\sqrt{3}}{4\pi} \sim 0.5865$ for all 2-dimensional RGGs in the Euclidean space (Dall and Christensen, 2002), and their assortativity tends to the same value (Antonioni and Tomassini, 2012; Barnett et al., 2007).

RGG networks thus possess some of the properties associated with social networks (strong clustering, positive assortativity, spatiality), but lack others (short characteristic path lengths, strong community structure, long-tailed degree distributions). The REDS model (Antonioni et al., 2014) is an extension to the standard RGG model that is able to readily exhibit all of these properties, save that, until now, REDS networks have not exhibitted the short characteristic path lengths characteristic of small world social networks.

In a REDS network, in addition to the standard RGG constraint that each edge must be shorter than a threshold distance, R, all edges impose an energy cost on the nodes that they connect, and the total cost of an individual node's edges may not exceed some finite threshold value, E. Each of a node's connections costs an amount of energy proportional to its Euclidean length, D. Moreover, the cost of an edge linking two nodes diminishes with the number of neighbours that the two nodes have in common, with the strength of this "synergy" effect governed by a parameter, S. Networks are constructed by assigning legal edges at random until no more edges can be afforded.

The resulting REDS networks resemble real social networks in several respects. They are spatially embedded, have high clustering, positive degree correlation, skewed degree distributions, and strong community structure. However, as mentioned above, standard REDS networks are not small worlds. While they are highly clustered, they have relatively long characteristic path lengths, like lattices. In order to model social processes such as the flow of information or disease in a structured population, it is desirable to employ a random network model that exhibits tuneable analogues of all the major properties of real social networks. Consequently, here we will first explore how readily "small world REDS" can be constructed through a Watts-Strogatz-style rewiring process (Watts and Strogatz, 1998), before proceeding to explore dynamics on REDS networks for the first time.

Small World REDS Networks

The REDS model comprises four components:

1. **R**each: an undirected edge, ij, between a pair of nodes, i and j, may only exist if the Euclidean distance between them, D_{ij}, is less than their "social reach", R.

2. **E**nergy: each node, i, has a finite quantity of "social energy", E, that may be spent on maintaining its edges.

3. **D**istance: the cost, c_{ij}, of edge ij is proportionate to the Euclidean "social distance", D_{ij}, between i and j.

4. **S**ynergy: the cost, c_{ij}, of edge ij varies inversely with the number of network neighbours that i and j share, k_{ij}. This effect is parameterised using $0 \leq S \leq 1$[2].

More explicitly, the cost of each edge is calculated as:

$$c_{ij} = \frac{D_{ij}}{1 + Sk_{ij}},$$

where k_{ij}, the number of neighbours shared by i and j, is the cardinality of the intersection between the set of i's neighbours and the set of j's neighbours.

The central intuition of *synergy* within the REDS model is exemplified as follows. Maintaining relationships with two neighbours that are themselves connected tends to be cheaper than maintaining the same relationships when the two neighbours are not connected to each other. In the former situation, direct interaction with one neighbour effectively involves an element of indirect interaction with the other (through gossip, chance encounters, group gatherings, etc). In more general terms, this is a local network effect that represents the potential for synergetic or catalytic interactions between the system elements.

[2] The upper bound on S may appear arbitrary, but $S > 1$ would imply that the total cost to a node within a fully connected clique of maintaining its edges would be less than the average distance betweene it and a single neighbour, which we claim to be physcically unrealistic.

Here we construct REDS networks as follows: 1) assign each of N nodes a random location in the unit square, 2) pick a random node, i, 3) pick a random node j such that $D_{ij} < R$ and j is not already a neighbour of i, 4) if both i and j can afford an edge between them, add it to the network, 5) repeat steps 3) and 4) until no more edges can be added. For a full account of the structural properties of REDS networks, see (Antonioni et al., 2014).

To create small world REDS networks we employ a random rewiring protocol adapted from Watts and Strogatz (1998): for each edge ij in the original network, with probability p, remove it and replace it with a new undirected edge kl, where k and l are randomly chosen nodes (Newman, 2010, p.555).

Results

An example REDS network, a rewired REDS network and a RGG with approximately equivalent mean degree are depicted in figure 1. Note that introducing even a very small amount of random rewiring significantly depresses the characteristic path length of the REDS network, but that it does not impact strongly on the clustering or community structure of the network. Throughout the paper, as a null model against which to compare the results for REDS networks, we will use RGGs with the same number of nodes and a threshold, R, that results in approximately equal average degree. Figure 1 shows that such an RGG tends to have less evident community structure, but a somewhat higher average clustering coefficient and also a longer characteristic path length.

Figure 2 depicts a systematic comparison between the effect of random rewiring on REDS networks and RGGs. As the probability of edges being rewired (p) increases, path length falls sharply for both classes of network, whereas clustering falls more slowly, ensuring that a small amount of rewiring can create small world networks where path lengths are relatively low and clustering is relatively high. As p tends to unity, both classes of network converge on fully randomised networks with the same values for clustering and path length.

Figure 3 quantifies the strength of the small world effect, demonstrating that it peaks at similar values of p for both classes of network, with the effect being felt more strongly for RGGs than for REDS networks.

Contagion on REDS Networks

To explore dynamics on REDS networks, we employ a standard SIS (Susceptible-Infected-Susceptible) model of contagion, a common extension of the classic Susceptible-Infected (SI) model, with the important addition of allowing for recovery and reinfection. It is widely used to model diseases that confer no or limited immunity (Newman, 2010).

Initially, a set of n randomly chosen nodes are set to be *infected*. The remainder are *susceptible*. A contagion process

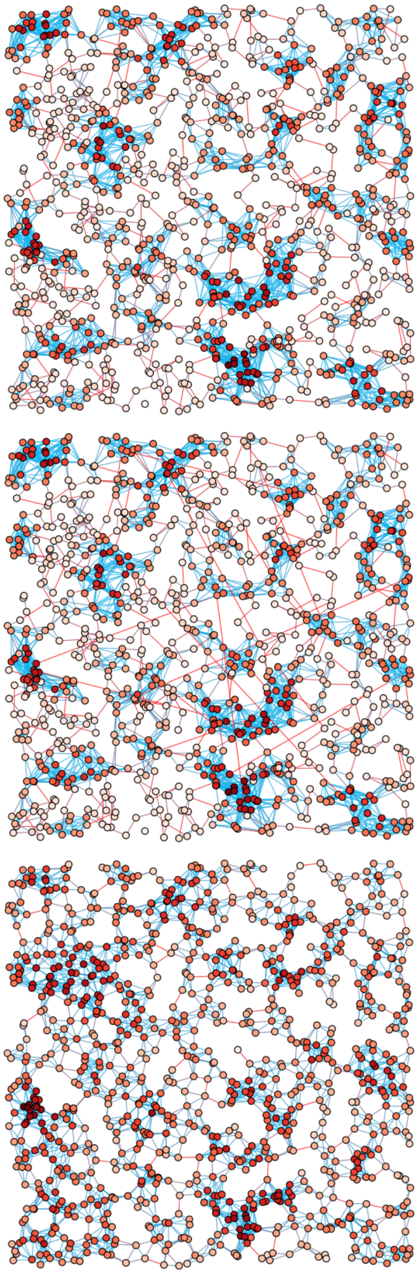


Figure 1: Example networks. Red nodes have higher degree and red edges are more expensive. *Top:* REDS network ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$, non-toroidal boundaries, mean degree: 8.4) exhibiting strong clustering (0.471), evident community structure, and relatively long characteristic path length (13.2). *Middle:* The same network subject to a small amount of random rewiring ($p = 0.005$). Path length falls by $\approx 26\%$ (to 9.71) and clustering by less than 1% (to 0.467). *Bottom:* RGG with equivalent degree ($N = 10^3$, $R \approx 0.05$, mean degree: 8.4). Clustering (0.6) and characteristic path length (14.15) are higher than for the REDS network, and respond similarly to random rewiring ($p = 0.005$ causes reductions of 36% and 2%, respectively).

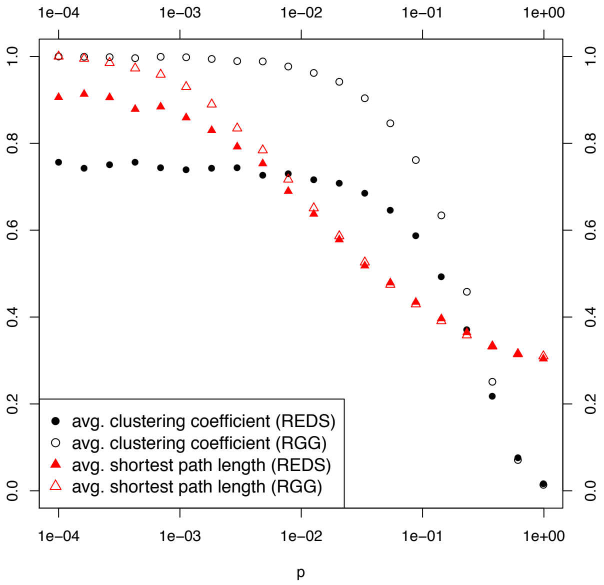


Figure 2: Curves demonstrating the effect of random rewiring on the average clustering coefficients and average shortest (i.e., characteristic) path lengths of REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$) and RGGs with approximately equivalent degree ($N = 10^3$, $R = 0.05$). Each data point is the average of 100 networks. All values are normalised w.r.t. those of unrewired RGGs. When $p = 0$, REDS networks have lower clustering and shorter path lengths than RGGs, but increasing p produces qualitatively equivalent effects on both measures.

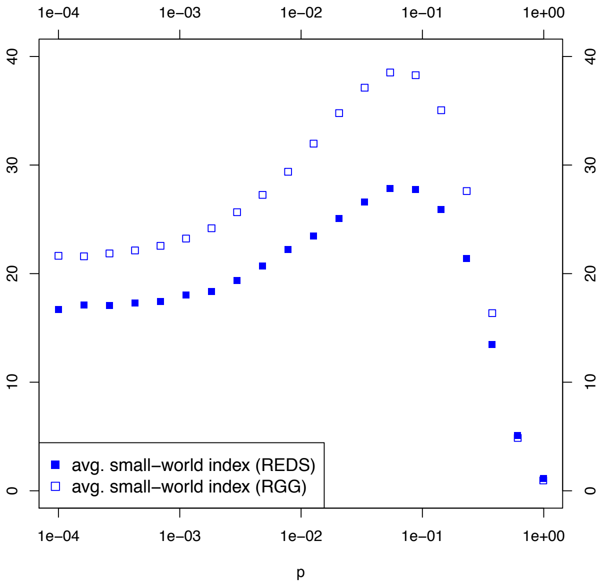


Figure 3: Curves demonstrating the strength of the small world effect brought about by randomly rewiring REDS networks and RGGs (parameters as figure 2). A network's small world index is calculated as $S = \frac{C/C(p=1)}{\lambda/\lambda(p=1)}$, where C and λ are the average clustering coefficient and characteristic path length of the network, respectively, while $C(p = 1)$ and $\lambda(p = 1)$ are the equivalent values calculated for a fully rewired variant of the network. The effect peaks at around the same rewiring probability for both classes of network, but is weaker overall for REDS networks.

then updates the status of each node synchronously for a sequence of discrete time steps during each of which at most one state change is allowed per node. Explicitly, at each time step, t: each node that was susceptible at time step $t - 1$ becomes infected with probability $1-(1-\beta)^{\gamma}$, where γ is the number of its neighbours that were infected at time step $t-1$ and $0 < \beta < 1$ is the infection's transmission probability; each node that was infected at time step $t-1$ reverts to being susceptible with probability μ.

Recovery from infection has an important consequence: the number of infected nodes must always be less than the total population. Hence, rather than an outbreak infecting all of the nodes on a network, the system will eventually *stabilize* when the rates at which new infections arise and infected nodes recover will be exactly equal, resulting in a steady fraction of the population exhibiting the infection (Newman, 2010).

Each finite population exhibits an epidemic threshold, i.e., a ratio between the transmission probability β and the recovery probability μ, separating endemic and extinction behaviour. Gomez et al. (2010) proposed a numerical methodology for calculating this threshold, β_c (dubbed critical β), based on dividing the recovery probability by the largest eigenvalue of the adjacency matrix (Λ_{max}):

$$\beta_c = \frac{\mu}{\Lambda_{max}}$$

This provides a quick and easy numerical indicator which, when it coincides with the output of simulations, indicates that the assumptions of the theoretical framework were not violated. Where $\beta < \beta_c$, the proportion of infected nodes will go to zero, and we will describe the outbreak as having failed. Where $\beta > \beta_c$ the system will stabilize with a positive number of infected nodes and we will describe the outbreak as having reached an endemic state. Moreover, this expression implies that changing the value of μ only results in a quantitative modification of β_c without influencing the actual dynamics of the system qualitatively. Therefore, when simulating an epidemic outbreak the value of μ is typically fixed at unity (Barrat et al., 2008).

Results

Here we report the results of SIS processes run on standard REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$, toroidal boundary conditions), the same REDS networks after suffering some degree of rewiring as described in the previous section, and RGGs that serve as a baseline for comparison, having parameters that give rise to networks with approximately the same average degree ($N = 10^3$, $R = 0.05$).

Due to the heterogeneous nature of our networks, the choice of the initial number of infected nodes can strongly influence the outcome of the simulations. Values reported in the literature range between 0.1% and 50% (Ferreri et al., 2014; Pastor-Satorras and Vespignani, 2001). Following Gomez et al. (2010), we chose to infect 5% (i.e., 50) of the network nodes initially. At lower values, many outbreaks may fail due to topological idiosyncrasies of the randomly chosen initial infected sites.

Each outbreak was simulated using the following protocol:

- Infect $n = 50$ unique, randomly selected nodes
- Run the SIS process for a minimum of 300 time steps and a maximum of 10^4 time steps (enough time for the infection to stabilize around its endemic state)
- An infection outbreak is deemed to have stabilized when the median number of infected nodes over the last 100 time steps is between the 49.5$^{\text{th}}$ percentile and 50.5$^{\text{th}}$ percentile of the previous 100 time steps.

To understand the effect of rewiring on REDS networks, we evaluated trends in β_c, the number of infected nodes after stabilization, and the percentage chance of outbreak failure/survival. Epidemic thresholds observed in simulation agree closely with the expected values.

In the absence of any random rewiring, the β_c value of a REDS network was determined to be lower than that of an equivalent RGG (figure 4). The net effect of randomly rewiring REDS networks is a monotonic increase in the value of β_c. In other words, the introduction of randomly rewired shortcuts and, more importantly, the resultant progressive loss of an organized community structure *reduces* the ease with which an outbreak achieves endemic stability. This can be seen in figure 4 and is confirmed numerically in figure 5.

If we classify this situation in terms of the simple question "does the simulation run survive or die out?", and express the answer as a percentage of total runs, we obtain figure 6. This classification trades off the ability to separate an endemic state from an epidemic breakout in favour of a clearer separation between the absorbing disease-free regime and the endemic regime, and clearly displays the increased infectiousness required by diseases on networks with higher levels of random rewiring.

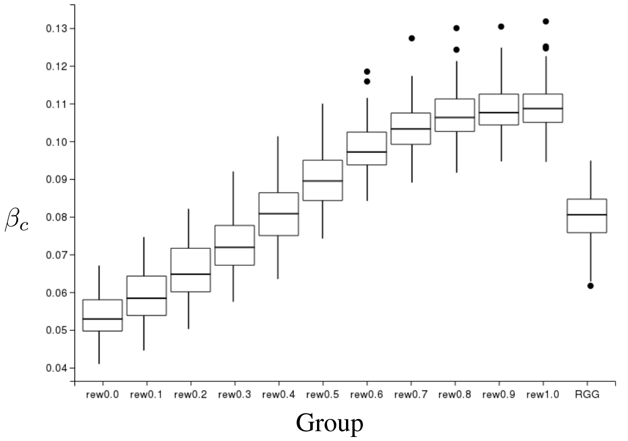


Figure 4: Variation in β_c for SIS contagion on REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$) subjected to different probabilities of random rewiring. RGGs with approximately equivalent degree ($N = 10^3$, $R = 0.05$) are plotted as a baseline. For all outbreaks, the initial number of infected nodes was $n = 50$ and the recovery rate was $\mu = 1$. Each box plot represents 100 independent outbreaks simulated on each of 100 networks.

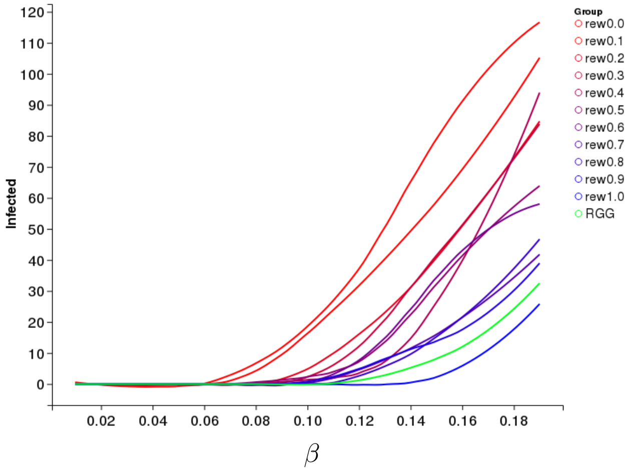


Figure 5: The mean number of infected nodes after stabilization for SIS contagion on REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$) subjected to different probabilities of random rewiring. RGGs with approximately equivalent degree ($N = 10^3$, $R = 0.05$) are plotted as a baseline. For all outbreaks, the initial number of infected nodes was $n = 50$ and the recovery rate was $\mu = 1$. Data plotted represents 100 independent outbreaks simulated on each of 100 networks for each value of $\beta \in [0, 0.01, 0.02, \ldots]$.

Overall, these results indicate that REDS networks are more susceptible to contagion than equivalent RGGs. Although random rewiring improves their ability to resist infection, a significant amount is required before they are as

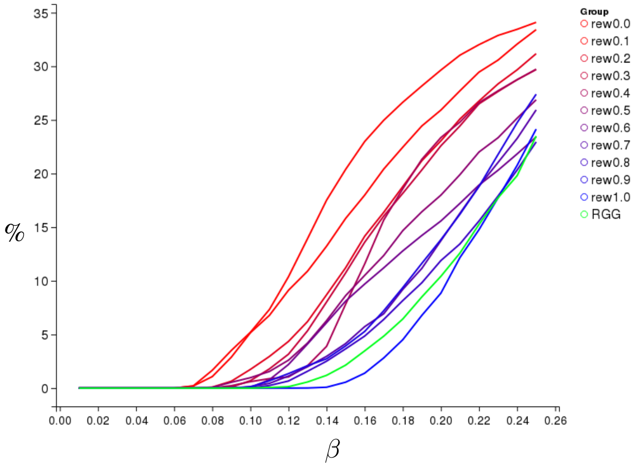


Figure 6: Percentage of SIS outbreaks that do not fail on REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$) subjected to different probabilities of random rewiring. RGGs with approximately equivalent degree ($N = 10^3$, $R = 0.05$) are plotted as a baseline. For all outbreaks, the initial number of infected nodes was $n = 50$ and the recovery rate was $\mu = 1$. Each smoothed curve represents data from 100 independent outbreaks simulated on each of 100 networks for each value of $\beta \in [0, 0.01, 0.02, \ldots]$.

resistant as equivalent RGGs (e.g., $p \gtrapprox 0.4$ in figure 4). Before discussing which properties of REDS networks are responsible for these results, we will explore a contagion dynamic that does not feature recovery and reinfection.

Selection Pressure on REDS Networks

Whereas the previous section explored a probabilistic contagion process with recovery, we now consider a spreading process without recovery, described in terms of the spread of a beneficial mutant strategy within a population.

When designing an artificial evolutionary system, researchers pay considerable attention to the choice of genetic encoding, parameter values for the genetic operators, and the selection mechanism. Darwin, however, realized that the "populations" that he observed also had spatial structure that influenced their population dynamics. For instance, species isolated on islands evolved differently from their counterparts living in more open environments. Geographical separation is a factor that shapes evolution.

The same principles apply to artificial evolution (Bäck, 1994). In evolutionary systems, topological population structure plays an important role in influencing the dynamical processes taking place. The simultaneous study of the structure and the dynamics of a given system is thus crucial. Structured populations are defined as populations in which any given individual has its own neighborhood, which is generally much smaller than the size of the population (Giacobini et al., 2005, 2004).

Changing the spatial aspect of an evolutionary algorithm influences the selection operator dynamics. In fact, the variation and mutation mechanisms operate on the individuals previously selected, whereas selection is directly influenced by the pool within which each individual is assessed, which is in turn determined by the topology of the population. In the artificial evolutionary environment, the selection mechanism is a key factor in the dynamics of the system, and different selection schemes have been characterized by the selection pressures they induce, usually described in terms of the takeover time. This is defined as being the time it takes for a single best individual to take over the entire population. It can also be seen as a simplified infection process, where an individual is infected when it is replaced by a copy of the best individual and where infected individuals cannot recover but instead stay infected forever. Takeover time can be estimated experimentally by measuring the propagation of the best individual's strategy under the sole effect of selection, without any variation operator. A shorter takeover time indicates a higher selection pressure and thus a more exploitative algorithm. Longer takeover times indicate lower selection pressure and a more exploratory algorithm.

Results

In order to explore the effect on selection pressure of rewiring REDS networks, we experimentally determine how the takeover time of a single best individual varies with network topology. Again, we explore the full spectrum of rewiring probabilities for REDS networks, $p \in [0, 0.01, 0.02, \ldots, 1]$, and also for a set of RGGs with approximately equivalent average degree. We generate one hundred different networks for each value of p, and also a set of one hundred RGG networks. For each network we repeat the takeover process 10,000 times starting with a randomly chosen initial best individual.

The vertices of each network represent a population of Boolean individuals (where '1' indicates the presence of the best genotype, and '0' the presence of some inferior genotype). The edges linking vertices represent the topology over which selection may operate. We analyze the propagation of a single best individual over the entire network using a simple synchronous discrete process that employs a binary tournament for the selection of individuals.

At initialization, all individuals in a population (equivalently: all nodes on the network) are set to '0' except for a single randomly chosen "best individual" which is set to '1'. Over a finite number of at most 100 time steps, the state of each individual is determined by selecting a single opponent among its neighbors. If the opponent's state is "1", the current individual will adopt that state, otherwise, its state remains unchanged. Thus only the genotype of the best individual propagates through the population. At each time step, we record the fraction of the population with the best genotype. The time step at which this value reaches unity is

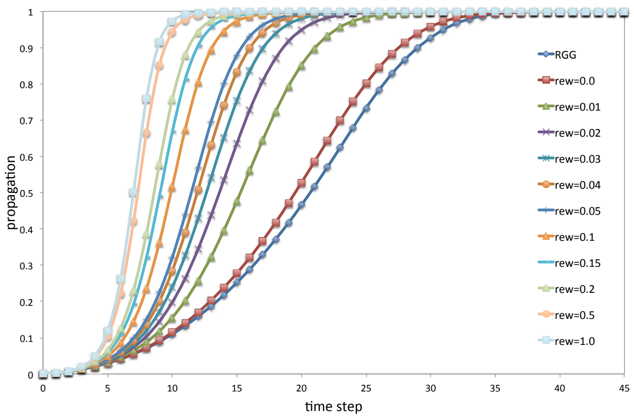


Figure 7: Propagation of the "Best Individual" over REDS networks ($N = 10^3$, $R = 0.08$, $E = 0.124$, $S = 1.0$) subjected to different probabilities of random rewiring, and over RGGs with approximately equivalent degree ($N = 10^3$, $R = 0.05$).

the takeover time.

Figure 7 represents averaged propagation over time. Each curve is the averaged result of 10,000 independent runs for one hundred networks. To increase the readability of the figure, we have selected a representative subset of rewiring probabilities. Unrewired REDS networks have a slightly faster takeover time than RGGs and hence a slightly higher selection pressure. As the probability of randomly rewiring REDS increases, the selection pressure increases monotonically—rapidly at first and then more slowly. In the next section we will consider why contagion processes on REDS networks respond to rewiring in the way that they do.

Discussion

We have explored the behaviour of two different contagion processes on REDS networks and on REDS networks that have been subjected to random rewiring. The first (SIS) process involved recovery and reinfection while the second (binary tournament EA) did not.

Before considering the effect of rewiring, we can first observe that REDS networks were more susceptible to contagion of both kinds than equivalent RGGs. For the SIS model, outbreaks were more able to arise and survive on REDS networks than on equivalent RGGs. For the selection pressure simulations, REDS had slightly faster takeover times than equivalent RGGs.

It is known that for diseases from which it is possible to recover and be reinfected (for the SIS results reported here $\mu = 1$ means that recovery is automatic) the persistence of the disease in a population is much less likely if that population is well-mixed than if it is more structured (Bolker and Grenfell., 1996). The lack of clustering within a well-mixed population encourages an infection to diffuse rapidly away from the original site of the outbreak and ensures that individuals have little chance of becoming reinfected. Conversely, a population with strong clustering resulting from the presence of local structure ensures that infection spreads to individuals who are able to reinfect one another should they recover, enabling an outbreak to become established in a local portion of the population before spreading.

However, our results suggest that it is not merely the strong clustering in REDS networks that is favouring outbreaks, since the clustering of the equivalent RGGs is higher yet they are more resistant to outbreaks. Rather, it is the stronger community structure of REDS networks that makes them more vulnerable to SIS outbreaks than RGGs. Randomly rewiring REDS networks erodes both clustering and community structure, preventing outbreaks from establishing themselves in a tightly connected local sub-population. However, this effect is gradual, and rewiring must be very prevalent before a REDS network becomes as resistant to outbreaks as an equivalent RGG (p larger than around 0.4 for the results reported here). Consequently, rewired REDS networks that maximise the small world index (e.g., $p \approx 0.01$) are still significantly more vulnerable to disease than equivalent RGGs.

For the second contagion process explored here (takeover of a beneficial mutant on a network via binary tournaments) REDS have an advantage over equivalent RGGs stemming from their shorter characteristic path lengths. Since there is no recovery/reinfection in this spreading process, contagion is not aided by local community structure, but is accelerated by short characteristic path lengths which ensure that one location on the network is only a small number of hops from any other, reducing the diameter of the network.

The increase in selection pressure (i.e., takeover speed) that results from increasing the amount of random rewiring applied to REDS networks is due to the introduction of shortcuts across the networks which shorten their characteristic path lengths and thereby facilitate diffusion.

In summary, on REDS networks a contagion process with recovery/reinfection is *inhibited* by the introduction of random rewiring, whereas one without recovery/rewiring is *exacerbated* by it. Consequently, by comparison with standard RGGs small world REDS networks will be highly susceptible to both kinds of contagion, since they combine strong community structure and relatively short characteristic path lengths.

Conclusion

In this paper we have explored simple dynamic contagion processes operating over a class of spatially embedded, community structured networks. By introducing a small amount of random rewiring we have been able to demonstrate the existence of a small world regime for REDS networks. We have shown that the strong community structure in REDS networks makes them particularly vulnerable to outbreaks of diseases that feature recovery and reinfection. When REDS networks are randomly rewired to the extent that they exhibit a strong small world effect, the consequent erosion of community structure is not enough to protect them from such outbreaks of infection. They remain more vulnerable than standard RGGs. However, since this rewiring does significnatly reduce their characteristic path length, it does accelerate the spread of any contagion that does not feature recovery/reinfection. These results suggest that real-world social networks, i.e., small world networks that are spatially constrained and subject to the influence of some local synergetic network effect, will tend to be more susceptible to a range of different contagion dynamics than was previously thought.

Acknowledgments: Bullock was supported by EPSRC grant EP/H021698/1. Darabos and Moore were supported by NIH grants R01 LM009012 and LM010098. Giacobini acknowledges local research funding from the University of Torino.

References

Antonioni, A., Bullock, S., and Tomassini, M. (2014). REDS: An energy-constrained spatial social network model. In Lipson, H., Sayama, H., Rieffel, J., Risi, S., and Doursat, R., editors, *Proceedings of the Fourteenth International Conference on Artificial Life*, pages 368–375. MIT Press, Cambridge MA.

Antonioni, A. and Tomassini, M. (2012). Degree correlations in random geometric graphs. *Physical Review E*, 86:037101.

Bäck, T. (1994). Selective pressure in evolutionary algorithms: A characterization of selection mechanisms. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, volume 1, pages 57–62. IEEE Press, Piscataway, NJ.

Barnett, L., Di Paolo, E., and Bullock, S. (2007). Spatially embedded random networks. *Physical Review E*, 76(5):056115.

Barrat, A., Barthélemy, M., and Vespignani, A. (2008). *Dynamical processes on complex networks*. Cambridge University Press, Cambridge, UK.

Barthélemy (2011). Spatial networks. *Physics Reports*, 499:1–101.

Bartlett, S. and Bullock, S. (2014). Natural convection of a two-dimensional Boussinesq fluid does not maximize entropy production. *Physical Review E*, 90(2):doi:10.1103/PhysRevE.90.023014.

Boerlijst, M. C. and Hogeweg, P. (1991). Spiral wave structures in pre-biotic evolution: Hypercycles stable against parasites. *Physica D*, 48:17–28.

Boguñá, M., Pastor-Satorras, R., Díaz-Guilera, A., and Arenas, A. (2004). Models of social networks based on social distance attachment. *Physical Review E*, 70:056122.

Bolker, B. M. and Grenfell., B. T. (1996). Impact of vaccination on the spatial correlation and persistence of measles dynamics. *Proceedings of the National Academy of Sciences USA*, 93(22):12648–12653.

Buckley, C. L., Bullock, S., and Barnett, L. (2010). Spatially embedded dynamics and complexity. *Complexity*, 16(2):29–34.

Bullock, S., Barnett, L., and Di Paolo, E. (2010). Spatial embedding and the structure of complex networks. *Complexity*, 16(2):20–28.

Bullock, S. and Buckley, C. L. (2009). Embracing the tyranny of distance: Space as an enabling constraint. *Technoetic Arts*, 7(2):141–152.

Dall, J. and Christensen, M. (2002). Random geometric graphs. *Physical Review E*, 66:016121.

Di Paolo, E. A. (2000). Ecological symmetry breaking can favour the evolution of altruism in an action-response game. *Journal of Theoretical Biology*, 203:135–152.

Ferreri, L., Bajardi, P., Giacobini, M., Perazzo, S., and Venturino, E. (2014). Interplay of network dynamics and heterogeneity of ties on spreading dynamics. *Physical Review E*, 90:10.1103/PhysRevE.90.012812.

Giacobini, M., Alba, E., Tettamanzi, A., and Tomassini, M. (2004). Modeling selection intensity for toroidal cellular evolutionary algorithms. In Deb, K., Poli, R., Banzhaf, W., Beyer, H.-G., Burke, E., Darwen, P., Dasgupta, D., Floreano, D., Foster, J., Harman, M., Holland, O., Lanzi, P., Spector, L., Tettamanzi, A., Thierens, D., and Tyrrell, A., editors, *Genetic and Evolutionary Computation – GECCO 2004*, pages 1138–1149. Springer, Berlin Heidelberg.

Giacobini, M., Tomassini, M., Tettamanzi, A., and Alba, E. (2005). Selection intensity in cellular evolutionary algorithms for regular lattices. *IEEE Transactions on Evolutionary Computation*, 9(5):489–505.

Gomez, S., Arenas, A., Borge-Holthoefer, J., Meloni, S., and Moreno., Y. (2010). Discrete-time markov chain approach to contact-based disease spreading in complex networks. *Europhysics Letters*, 89:38009.

Newman, M. E. J. (2010). *Networks: An Introduction*. Oxford University Press, Oxford.

Pastor-Satorras, R. and Vespignani, A. (2001). Epidemic dynamics and endemic states in complex networks. *Physical Review E*, 63(6):066117.

Penrose, M. (2003). *Random Geometric Graphs*. Oxford University Press, Oxford, UK.

Serrano, M., Krioukov, D., and Boguná, M. (2008). Self-similarity of complex networks and hidden metric spaces. *Physical Review Letters*, 100:078701.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, 393:440–442.

Wong, L. H., Pattison, P., and Robins, G. (2006). A spatial model for social networks. *Physica A*, 360(1):99–120.

zu Erbach-Schoenberg, E., Bullock, S., and Brailsford, S. (2014). A model of spatially constrained social network dynamics. *Social Science Computer Review*. 0894439313511934.

A Passive Mechanism for Goal-Directed Navigation using Grid Cells

Vegard Edvardsen

Department of Computer and Information Science,
Norwegian University of Science and Technology, Trondheim, Norway
vegarded@idi.ntnu.no

Abstract

As more is becoming understood about how the brain represents and computes with high-level spatial information, the prospect of constructing biologically-inspired robot controllers using these spatial representations has become apparent. Grid cells are particularly interesting in this regard, as they provide a general coordinate system of space. Artificial neural network models of grid cells show the ability to perform path integration, but important for a robot is also the ability to calculate the direction from the current location, as indicated by the path integrator, to a remembered goal. Present models for goal-directed navigation using grid cells have used a *simulating* approach, where the networks are required to actively test successive locations along linear trajectories emanating from the current location. This paper presents a *passive* model, where differences between multi-scale grid cell representations of the present location and the goal are used to calculate a goal-direction signal directly. The model successfully guides a simulated agent to its goal, showing promise for implementing the system on a real robot in the future. Some possible implications for neuroscientific studies on the goal-direction signal in the entorhinal/subicular region are briefly discussed.

Introduction

Results from neuroscience are gradually uncovering the neural basis for navigation, as cell types such as place cells and grid cells, first discovered in the hippocampal region of rats, have been shown to represent high-level features of the animal's spatial environment. These findings offer the prospect of beginning to understand how the brain computes and represents abstract cognitive features. Inspired by these advances, the basis for this project has been to devise and implement a neural model to enable a robot to find its way to a previously visited goal location using these neural representations of space known from the brain. Through crafting these models, we hope to gain insights into how these spatial representations might be utilized for navigational purposes by neural systems, artificial and real alike.

The first evidence of spatially responsive cells in the rat hippocampus came with the discovery of place cells, which were seen to respond at distinct locations in the environment (O'Keefe and Dostrovsky, 1971). However, place cells do not appear to encode any metric relations, such as distances and angles (Spiers and Barry, 2015). The place cell representation by itself is thus not sufficient to be able to navigate between arbitrary locations, because it does not offer any means to calculate the direction of travel from one place cell's firing location to that of another. Grid cells, discovered later in the neighboring *entorhinal cortex* (Hafting et al., 2005), offer a possible solution, as the grid cell system can be seen as a general spatial coordinate system. Given the grid cell representations of two locations it is possible to compute the distance and angle between them, thus providing the needed metric of space.

Grid cells are thought to update their firing activity based on self-motion information, in other words to perform path integration/dead reckoning (McNaughton et al., 2006). However, for a path integrator to be fully useful for navigational tasks, an agent should be able to use this information to find its way back to previously visited locations. In this paper we shall see how the activity of path integrating grid cell networks can be used to guide a simulated agent toward a remembered goal location.

Related Work

Path integration is the basis for several computational models of grid cells, the collection of which can roughly be divided into two major categories; oscillatory-interference models and attractor-network models (Giocomo et al., 2011). Several biologically-inspired models for navigation have used such models of grid cells (Milford and Schulz, 2014; Spiers and Barry, 2015). Often this has been for the purpose of position tracking, as in the bio-inspired robot navigation system "RatSLAM" (Milford and Wyeth, 2010), but in some cases grid cells have also been used for direction-finding.

Erdem and Hasselmo (2012) use a model with oscillatory interference-based grid cells to find directions to remembered goal-locations. The mechanism involves testing a number of "look-ahead probes" that trace out linear beams radially from the current location of the agent. Each of these probes orchestrate activity across the entire population of grid cells and place cells to make it appear as if the agent were actually situated at the tested coordinates. If any of the successive locations tested during a given look-ahead probe triggers a reward-associated place cell, the agent is impelled to travel in the specific direction of that probe. Kubie and Fenton (2012) show how a Hebbian learning mechanism between conjunctive grid cells can train the grid cell networks to be able to generate look-ahead trajectories similar to those suggested by Erdem and Hasselmo (2012). The authors propose that this is a "viable candidate for vector-based navigation". Common to these two approaches is the requirement for the model to explicitly test a wide range of different directions emanating from the current position, in order to expectedly trigger the goal-reward in some specific direction. In this sense, we can term these models *active* mechanisms for goal-directed navigation.

The model proposed in this paper goes the other way about the problem, by presuming that the grid cell-representation of the goal location is known beforehand. The current grid cell state and the goal grid cell state propagate through a pre-wired network of neurons that calculate the offsets between the two representations in order to generate a direction signal. This process can be constantly ongoing in the background, without requiring exclusive use of or otherwise interfering with any grid cell or place cell population for the purpose of "simulating" forward locations. We thus consider our model a *passive* mechanism for goal-directed navigation.

Background

Grid Cells are Organized in Modules

The name "grid cell" stems from the spatial activity patterns of these neurons; the cells are not active only within single spatial fields in the environment as the place cells are, but have a periodic pattern of activity that repeats at the vertices of a triangular tiling of the plane. The result is a hexagonal grid pattern, extending indefinitely throughout space, that can be characterized by the three properties of scale, rotation and phase—respectively the distance between two neighboring vertices of the grid pattern and the rotation and translation of the grid pattern compared against a frame of reference. A grid cell does not operate in isolation, but participates in a *module* of grid cells that share the same scale and rotation of their individual firing patterns (Stensola et al., 2012). The only distinguishing property between neurons within the same grid cell module is thus that of their phase (Figure 1).

Assuming that a sufficient number of grid cells belong to a given module, the module as a whole has the ability to encode a given set of 2D coordinates in a nearly continuous fashion. The limitation lies in the periodic nature of the grid cell pattern, in that the information carried by a grid cell module can only be interpreted relative to one specific hexagonal tile of the infinitely repeating pattern. A possible solution comes from the fact that the entorhinal cortex harbors grid cell modules of multiple different scales (Stensola et al., 2012). It is conceivable that the smaller-scaled grid cell modules represent space at a finer resolution than the larger-scaled ones, but with the sacrifice of having shorter "ranges of validity" due to the more rapid periodicity of the grid pattern. The activity of all modules taken collectively, however, ought to retain both the low-precision/long-range information of the larger-scaled modules as well as the high-precision/short-range information of the smaller-scaled modules. The utilization of this multi-scale mix of information is a key idea behind the model presented in this paper.

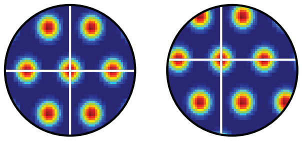


Figure 1: Idealized illustration of the activity of two different grid cells, as a function of the 2D location of the agent, in a cylindrical enclosure of e.g. two meters in diameter (top-down view). Red and blue colors indicate high and low firing rates, respectively. The two grid cells belong to the same grid module, as per the identical scale and rotation of their respective grid patterns, but are seen to have different phases, i.e. different offsets of these patterns, as indicated by the lines.

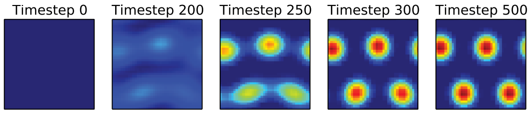


Figure 2: Spontaneous formation of a grid-like activity pattern in the neural sheet of an attractor-network grid cell module, due to random initial conditions and the recurrent connectivity.

Attractor-Network Models of Grid Cells

Attractor-network models of grid cells conceptualize their neurons as being laid out in a 2D neural sheet. Proximity between neurons in this sheet implies that the neurons should have similar phases of their grid patterns, not necessarily that the neurons would be co-located in the brain. The neurons are recurrently connected to each other, with a connectivity profile based on distances in the neural sheet, in such a way that grid-like patterns of activity will form in the network from random initial conditions (Figure 2).

These network patterns, which are the attractor states of the network, can then be made to shift around in the network in response to self-motion signals in order to perform path integration. Assuming these shifts consistently reflect the actual movements of the agent, the network pattern will over time become visible in the spatial activity patterns of *individual* neurons in the network. Attractor-models of grid cells thus have grid-like patterns both in their time-averaged spatial activity plots (Figure 1) and in their momentary network activity plots (Figure 2), and this is an important distinction to be aware of.

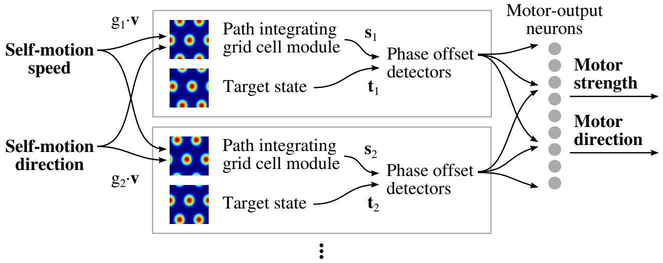


Figure 3: A schematic overview of the model.

Model

The main part of the model comprises a configurable number of *modules*, seen as the two rectangular blocks in the middle of Figure 3. Each module m consists of (a) a grid cell module, (b) a *target signal*, and (c) a network of *phase offset detectors*. The grid cell modules perform path integration on the incoming self-motion signal (composed of speed and direction), and output vectors of grid cell-activity $\mathbf{s}_m$ that are passed on to the corresponding networks of phase offset detectors. These phase offset detectors also receive a copy of the intended grid cell-activity vector $\mathbf{t}_m$ for the desired target location—the "target signal". The task of the phase offset detectors is to find the required direction of travel to make up for the offset in the grid patterns between the path integrator signal $\mathbf{s}_m$ and the target signal $\mathbf{t}_m$. The intended outputs of the model are a motor direction signal, giving the direction toward the target location, and a motor strength signal, indicating whether the agent has arrived at the target location or to keep going.

Multiple Modules with Different Spatial Scales

The model has these multiple parallel modules in order to utilize information from a variety of grid cell modules representing space at different scales; this will provide the direction-finding process with long-range/low-precision signals as well as short-range/high-precision signals. The different grid scales are achieved by modulating the velocity inputs to each grid cell module. The velocity signal to module m is multiplied by the gain factor g_m before reaching the grid cell network. Smaller gain factors will cause the path integrator to respond more slowly to the same velocity inputs, thus causing the grid to appear larger, and vice versa.

The path integrator model used in these simulations was found to respond acceptably to velocity inputs at least in the range from 0.1 m/s to 1.2 m/s. As the actual speed of the simulated agent was fixed to 0.2 m/s, the range of acceptable gain factors could then be determined to be $[g_{min}, g_{max}] = [0.5, 6.0]$. The model uses a geometric progression from g_{min} to g_{max} for the gain factors. Given a specific number of modules M to be used, the g_m values can then be calculated as

$$R = \sqrt[M-1]{g_{max}/g_{min}}, \qquad g_m = g_{min} \cdot R^{m-1}. \tag{1}$$

Path Integrating Grid Cell Modules

The path integrator modules are closely based on the attractor-network grid cell model by Burak and Fiete (2009), and the following formulas are based on their presentation of the model. Each grid cell module consists of a 2D sheet of neurons of size $n \times n$, where $n = 40$. The activation values of these $n^2 = 40^2 = 1600$ neurons are contained in a vector $\mathbf{s}$, fully representing the current state of the path integrator.

Each grid cell i receives recurrent inputs from all other neurons in $\mathbf{s}$. Let $\mathbf{x}_i$ be the neural sheet coordinates of neuron i. The weight from afferent neuron i' onto neuron i can then be calculated from the connectivity profile $rec(d)$ by letting d be the shortest distance between $\mathbf{x}_{i'}$ and $\mathbf{x}_i$ in the neural sheet, taking into consideration that connectivity may wrap around the N/S and W/E edges. The recurrent connectivity profile $rec(d)$ is a difference of Gaussians, seen as the inhibitory "doughnut" in Figure 4, top left. Specifically,

$$rec(d) = e^{-\gamma d^2} - e^{-\beta d^2}, \tag{2}$$

where $\gamma = 1.05 \cdot \beta$, $\beta = \frac{3}{\lambda^2}$ and $\lambda = 15$. λ approximately specifies the periodicity of the grid cell network, i.e. the number of neurons from one peak of activity to the next.

To express the update rule for grid cell i using vector notation, let $\mathbf{w}^{rec}_{\mathbf{c}}$ be the weight vector derived from the distance-to-weight-profile $rec(d)$ centered on the point $\mathbf{c}$ in the neural sheet. The update rule can then be described as

$$\tau \frac{ds_i}{dt} + s_i = f\left(\mathbf{s} \cdot \mathbf{w}^{rec}_{\mathbf{x}_i - \hat{\mathbf{e}}_{\theta_i}} + B_i\right), \tag{3}$$

solved for ds_i, where $dt = 10$ ms, $\tau = 100$ ms, $f(x) = max\,(0, x)$ and $\mathbf{s}$ is the vector of the activation values at the end of the previous timestep.

The center point $\mathbf{c}$ of the connectivity profile for efferent neuron i is here given as $\mathbf{x}_i - \hat{\mathbf{e}}_{\theta_i}$, i.e., there is an extra offset of $\hat{\mathbf{e}}_{\theta_i}$ in addition to $\mathbf{x}_i$ when positioning the connectivity profile for neuron i. The offset $\hat{\mathbf{e}}_{\theta_i}$ is the unit vector in the direction of θ_i, which in turn is the *directional preference* of neuron i. The directional preference is used to shift the activity pattern among the grid cells in response to asymmetrical velocity inputs. Preferences for each of the four cardinal directions are distributed among the neurons in each 2×2 block of neurons. Namely, the x, y coordinates of a neuron are used to calculate an index $(2 \cdot (y \bmod 2) + x \bmod 2)$ into the list $[W, N, S, E]$ to determine θ_i. In the absence of velocity inputs, the four distinct preference-offsets counterbalance each other to keep the activity pattern at rest in the network. During motion, however, the external input B_i to each neuron becomes velocity-tuned according to the directional preference of the neuron. This input is calculated as

$$B_i = 1 + g_m \alpha \hat{\mathbf{e}}_{\theta_i} \cdot \mathbf{v}, \tag{4}$$

where $\mathbf{v}$ is the movement velocity and $\alpha = 0.10315$ is a scaling constant specified by Burak and Fiete (2009).

Phase Offset Detectors

The vector of activation values $\mathbf{s}$ is passed on to a network of phase offset detectors. In addition to receiving the input vector $\mathbf{s}$ from the path integrating grid cell module, the phase offset detectors also receive a similarly-shaped vector $\mathbf{t}$ that represents the grid cell activity of the target location, i.e. a grid cell-encoding of the desired target coordinates.

Each phase offset detector j has an associated origin location $\mathbf{x}_j$ and a preference direction θ_j. The neuron is tuned to respond when an activity peak is near the origin location $\mathbf{x}_j$ in the neural sheet of the path integrator grid cell module ($\mathbf{s}$) and there *simultaneously* is an activity peak near the location $\mathbf{z}_j = \mathbf{x}_j + \delta \hat{\mathbf{e}}_{\theta_j}$ in the grid cell-encoded target-location-input ($\mathbf{t}$). Specifically, the activation value of phase offset detector j is calculated as

$$p_j = f\left(\mathbf{s} \cdot \mathbf{w}^{in}_{\mathbf{x}_j} + \mathbf{t} \cdot \mathbf{w}^{ex}_{\mathbf{z}_j}\right), \tag{5}$$

where $\mathbf{w}$ again refers to weight vectors derived from given connectivity profiles centered on given points in the neural sheet, but with new connectivity profiles in and ex. The path integrator inputs $\mathbf{s}$ are fully connected using the connectivity profile $in(d)$ centered at $\mathbf{x}_j$, while the target location inputs $\mathbf{t}$ are fully connected using the connectivity profile $ex(d)$ centered at $\mathbf{z}_j$. These connectivity profiles are defined as

$$in(d) = \eta \cdot \left(e^{-\beta d^2} - 1\right), \qquad ex(d) = e^{-\beta d^2}, \tag{6}$$

where $\eta = 0.25$. The offset length δ is set to be $\delta = 7$, in the neighborhood of half of λ.

The effect is to respond the most strongly when there is an offset of length δ in direction θ_j between the activity patterns in $\mathbf{s}$ and $\mathbf{t}$, given that the path integrator currently has activity in the vicinity of $\mathbf{x}_j$. An example situation is shown in Figure 4, where a phase offset detector with $\mathbf{x}_j = (20, 20)$ and $\theta_j = 45°$ receives inputs of favorable characteristics from $\mathbf{s}$ and $\mathbf{t}$.

In order for the network of phase offset detectors to work independently of the current location of network activity in the path integrator, there needs to be a sufficient number of phase offset detectors that sample different origin locations $\mathbf{x}_j$. Additionally, the network needs to sample a range of different preference directions θ_j. This is realized using two parameters S_θ and S_{xy} that respectively specify the number of directions sampled in the interval $[0, 2\pi)$ and the number of steps to use along each of the two dimensions of the neural sheet when sampling origin locations. The total number of phase offset detectors will then be $S_\theta \cdot S_{xy}^2$ per module.

Motor-Output Neurons

The activity from the phase offset detectors are aggregated in a set of *motor-output neurons*. Whereas the grid modules and phase offset detectors are instantiated separately for each module, the motor-output neurons comprise a common network receiving inputs from all of the modules. The number of motor-output neurons is the same as the number of sampled preference directions S_θ in the phase offset detector networks. The motor-output neurons sample the same directions as the phase offset detectors.

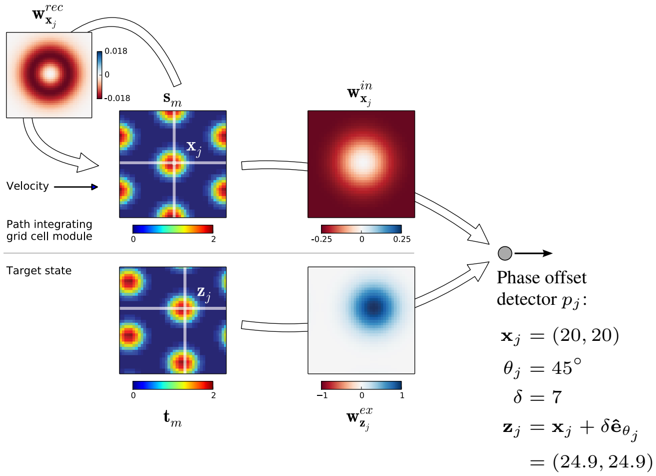


Figure 4: Example of a phase offset detector, p_j, showing the input networks $\mathbf{s}$ and $\mathbf{t}$ and the connectivity profiles with which these two networks are connected to p_j. Depicted matrices are 40×40.

The activity in each motor-output neuron is essentially the sum of the activity in all of the phase offset detectors that share preference direction with the motor neuron. An important detail, however, is that these contributions are weighted by the inverse of the gain factors of their respective modules. In other words,

$$u_k = \sum_m \left[g_m^{-1} \sum_j^{\theta_j = \theta_k} p_{m,j} \right], \tag{7}$$

where u_k is the activity of motor-output neuron k and θ_k is the preference direction of k. This weighting will give priority to the direction signals from the modules with low gain factors g_m, i.e. the modules where the quality of the path integration information is long-range-applicable but with low precision. As the agent gets closer to the target location, the intention is for these signals to fade off to sufficiently weak strengths so that the shorter-range, higher-precision signals will pick up in motor influence. The purpose is to achieve the trade-off of a long-range *and* high-precision signal.

To calculate the final motor-output signal $\mathbf{\Theta}$, the values of u_k are considered as vector contributions in the direction of θ_k, i.e. the vectors $u_k \cdot \hat{\mathbf{e}}_{\theta_k}$ are summed together, and this sum is then scaled to compensate for the variable number of inputs and their weighting. The final calculation is thus

$$\mathbf{\Theta} = \rho \cdot \sum_k u_k \cdot \hat{\mathbf{e}}_{\theta_k}, \quad \rho = \frac{1}{S_\theta \cdot S_{xy}^2 \cdot \sum_m g_m^{-1}}, \tag{8}$$

whereafter the angle of $\mathbf{\Theta}$ makes up the motor direction signal and the vector length becomes the motor strength signal.

Experiment Setup

Each experiment trial consists of a succession of stages, specifically (a) pattern formation in grid cell modules, (b) capture of path integrator states into target states, (c) the agent performing a random walk for T seconds, and (d) the agent attempting to return "home" to the target location.

At the beginning of the simulation, in order for the grid cell networks to form grid-like activity patterns, all s_i values are initialized randomly in the range $[0, 10^{-4})$ before the networks are then allowed to settle for 1000 timesteps (Figure 2). When this pattern formation process is done, the grid-like activity patterns will have been initialized to essentially random starting-coordinates. The model now copies these activity patterns $\mathbf{s}_m$ into the target state vectors $\mathbf{t}_m$. The m different target state vectors $\mathbf{t}_m$ henceforth remain unchanged for the rest of the trial, as a memory of the coordinates of the starting location ("home").

The agent then performs a random walk for a configurable duration of time T seconds. The time duration for a single iteration of the model has been set to be 10 ms, so there are 100 timesteps$/$s. During both the random-walk and the return-home stages, the agent moves with a constant speed of 0.2 m/s with only the movement direction changing. The random walk starts with a uniformly distributed random value from $[0, 2\pi)$ as the movement direction. At every timestep it is updated by adding a radian value from a normal distribution with $\mu = 0$, $\sigma = 1$.

After T seconds have elapsed, the return-home stage begins. The motor-direction output from the network is used to set the movement direction of the agent, whereas the motor-strength output is used as a termination criterion for determining when to end the trial. Three different termination criteria are used; (a) the motor-strength signal is less than 10^{-6}, (b) the return-home stage has lasted for at least a second and the straight-line distance to the point traversed one second ago is less than 0.01 m, or (c) the return-home stage has lasted $2 \cdot T$ seconds. Whichever termination criteria ends the trial, the straight-line distance to the starting location from the final stopping location is deemed the *error* of the trial. The favorable outcome is a low overall error value.

Parameter Search

A parameter search was conducted to find the best values for M, S_θ and S_{xy} to use for the rest of the experiments. An exhaustive test was performed on all combinations of values in the intervals $M \in [2, 6]$, $S_\theta \in [4, 32]$, $S_{xy} \in [5, 40]$. However, to penalize expensive solutions and to place an upper bound on the complexity of the solutions to be tested, a *synapse cost* C was calculated for each parameter combination. This value provides an estimate on the number of synapses in the model and consequently a rough estimate on the number of floating-point operations required to update the model (without optimization). C was calculated as $C = M \cdot \left(n^2\left(n^2+1\right) + 2n^2 \cdot S_\theta \cdot S_{xy}^2 + S_\theta \cdot S_{xy}^2\right)$, with the three terms representing the synapse cost to operate respectively a grid cell module, the phase offset detectors and the axons to the motor-output neurons. Only the combinations with $C < 10^8$ were tested, leaving 2685 combinations to test. For each combination, 100 trials with a random-walk duration of $T = 30$ s were performed and the mean error was reported (Figure 5a). The parameter combination $M = 4$, $S_\theta = 28$, $S_{xy} = 9$ was selected for further use (highlighted).

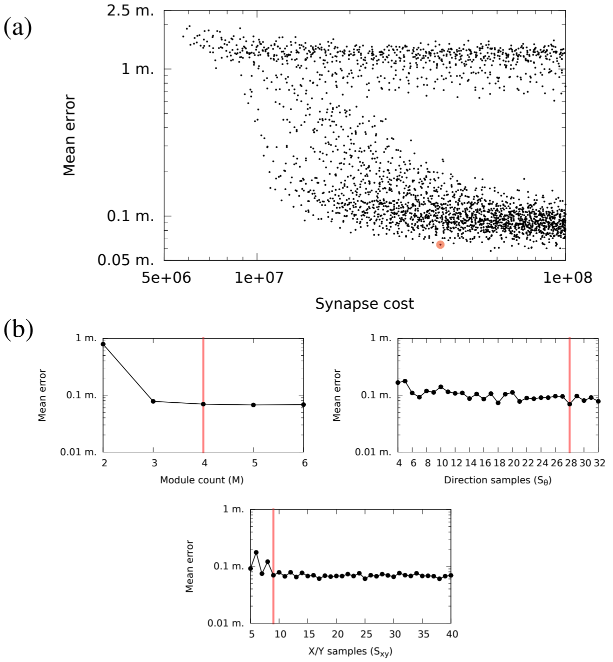


Figure 5: (a) Scatter plot of parameter combinations tested during the parameter search, with each dot showing the mean error over 100 runs for a given parameter combination, plotted against the respective synapse cost. (b) For each of the three parameters M, S_θ and S_{xy}, the effect of modifying that parameter from the chosen parameter combination ($M = 4$, $S_\theta = 28$, $S_{xy} = 9$; indicated by vertical lines) while leaving the other parameters unchanged. Mean error over 100 runs. The combination $M = 4$, $S_\theta = 28$, $S_{xy} = 9$ is represented by the same set of trials in all three figures.

To get a sense for how the individual parameters affect the outcome, new sets of runs were performed where each parameter in turn was changed within the defined intervals and evaluated over 100 trials, while the two other parameters were left unchanged (Figure 5b). M and S_{xy} seem to affect the results little above thresholds of respectively $M = 2$ and $S_{xy} = 8$. S_θ, on the other hand, appears to be more sensitive to the particular value to which it is assigned.

Implementation Details

All random values used by the implementation in this paper were generated using the Mersenne Twister pseudo-random number generator included with the C++11 standard library. The model and the simulator used single-precision floating-point values throughout.

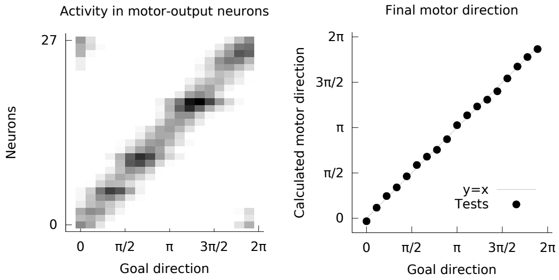


Figure 6: Example of network operation at locations a fixed radius $r = 0.5$ m away from the home location in given directions. Left: Momentary activity in the $S_\theta = 28$ motor-output neurons. Right: Final motor direction calculated from the motor-output neurons, plotted at the closest 2π period to the goal direction.

Results

Direction-Finding Ability

Two different examples of how the system operates in practice are presented in Figures 6 and 7. In the first example, the direction-finding ability of the model is tested at multiple points along a circle centered on the goal location. For each of the 18 uniformly spaced directions tested, the agent was driven a distance of 0.5 m in the opposite direction of the intended "goal direction" and allowed to settle for 250 timesteps before the motor outputs were examined. For each trial, the figure shows the recorded activity from all S_θ motor-output neurons as well as the motor-direction signal from the model. As evidenced by the figure, the model is able to accurately calculate the goal direction at this specific distance of $r = 0.5$ m.

Figure 7 demonstrates a full trial with both random-walk and return-home stages as described above. After a $T = 30$ s random walk, the agent successfully attempts a return to the home location. The figure includes the momentary activity of all of the grid cell modules ($\mathbf{s}_m$) both at the beginning and the end of the return-home stage. In each of these cases, the motor-neuron activity is also shown. The plots of $\mathbf{s}_m$ show possible interpretations of how the activity patterns might have shifted from the target state $\mathbf{t}_m$, which was also the initial state of the grid modules at $T = 0$ s. From the leftmost to the rightmost columns, the grid modules progress from long-range/low-precision to short-range/high-precision. The first, second and third modules show a correct assessment of the goal direction at $T = 30$ s, whereas the fourth module is "out of range" and in this case has an ambiguous response.

At $T = 37.1$ s, we see that the grid modules have aligned closely with the corresponding target states. The trial thus terminated because of the weak motor-strength signal, bringing the agent to a halt at a distance of 4.74 cm from the goal location, from an initial goal distance of 1.47 m at the end of the random walk.

Effect of Multiple Grid Modules

The effect of using multiple grid modules is further demonstrated in Figure 8, which shows, as a function of the distance to the goal, the strengths and errors of each module's contribution to the motor-output network when seen in isolation. To illustrate their relative influences, the signal strength of each module is also shown in terms of its ratio of the sum. Lastly, the final motor-direction error is shown overlaid on the direction-error plots from the individual modules.

For each module, there is a distinctive bell-shape in the strength curve as the tested radius approaches and recedes from the "optimal detection distance" of the module's offset detectors. The vicinity of the peak of the bell curve is also

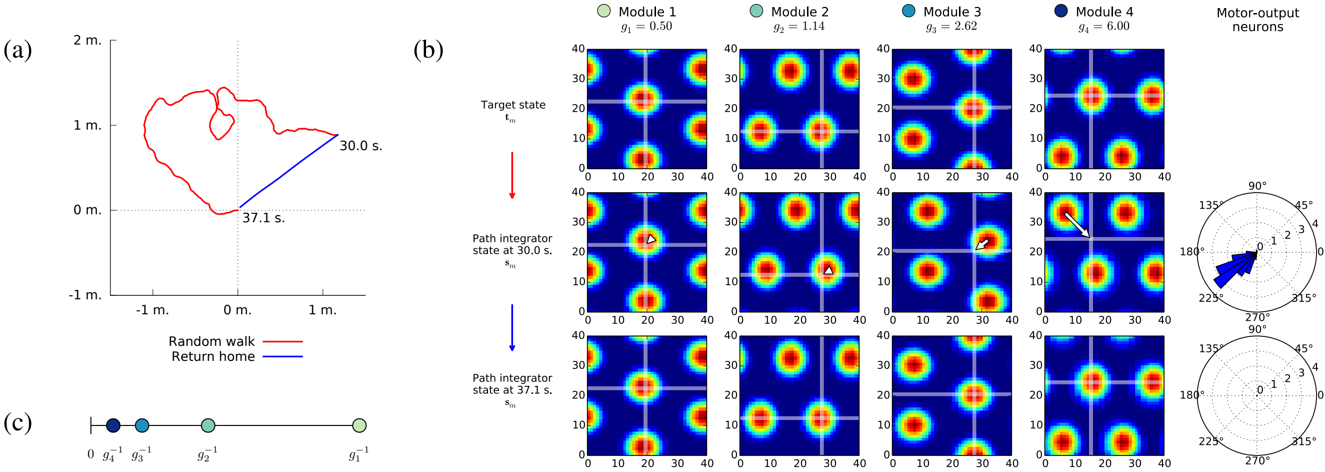


Figure 7: Example of network operation during various stages of a trial. (a) Trace of the trajectory followed during random-walk and return-home stages. (b) Top row: Target state. Rest: Momentary activity of the grid cell modules and motor-output neurons at various points in time. (The s_i and t_i values were arranged according to $\mathbf{x}_i$ and convolved with a uniform 2×2 filter to hide artifacts due to different preference-offsets. This procedure was used for all $\mathbf{s}$ and $\mathbf{t}$ visualizations in this paper.) (c) Plot of the inverses of the gain factors g_m used when the model is configured for four modules ($M = 4$), in order to illustrate the difference in the ranges of the modules.

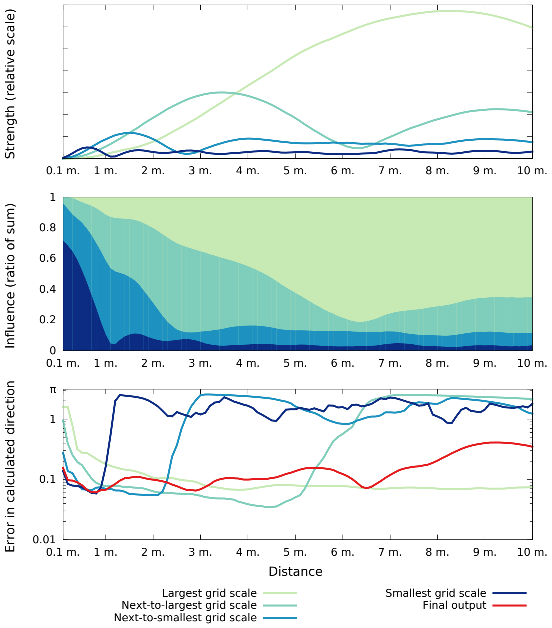


Figure 8: Behavior of the $M = 4$ different modules at increasing distances from the goal. The model was tested as in Figure 6, but r was varied at 0.1 m increments in the interval $[0.1 \text{ m}, 10 \text{ m}]$. For each tested radius the motor-output strengths and direction-signal errors are reported as the mean over 18 tested directions. In order to report values individually for each module, extra motor-output networks were instantiated such that each only received phase offset detector-inputs from one given module. For these plots, $\rho = 1$ in order not to cancel out the scaling differences.

where the module's direction-error is at a minimum. Past this region, the module abruptly becomes unreliable due to the periodicity of the grid cell signal. Because of the gain-based weighting of module contributions, however, one of the larger-scaled modules is able to overpower the contributions of the smaller-scaled modules and thus ensure that the final direction signal is still valid. As seen by the red line in the lower diagram, the final direction-output achieves a trade-off between range and precision not seen in any of the individual modules.

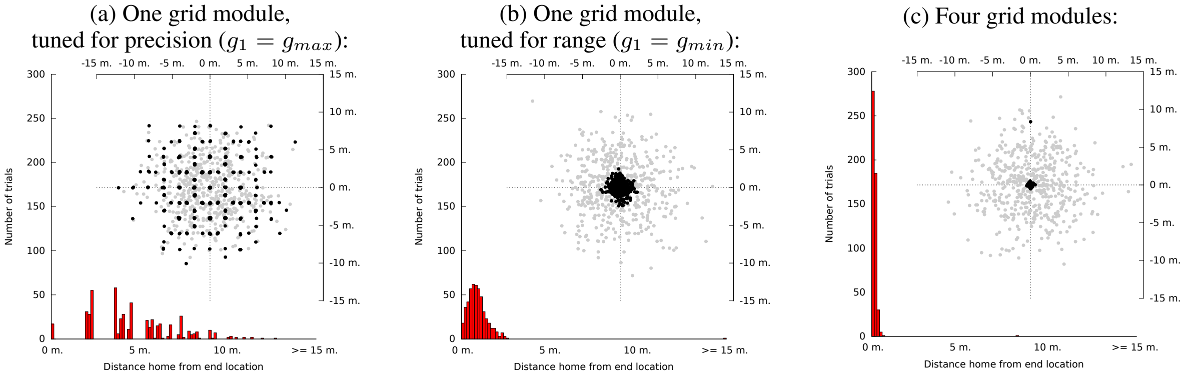
Figure 9 demonstrates the importance of this combination of precision and range information. The figure contains results from three different sets of 500 trials, each with $T = 180$ s. Whereas the rightmost diagram shows the results from trials with the default parameters ($M = 4$, $S_\theta = 28$, $S_{xy} = 9$), the two other diagrams only use one module ($M = 1$). The leftmost diagram has the gain factor set to $g_1 = g_{max}$, for rapid periodicity and short-range/high-precision signals, while the middle diagram has $g_1 = g_{min}$, i.e. tuned for long-range/low-precision signals.

The distributions of termination locations seen in the scatterplots confirm our expectations from the known qualities of the grid module signals. With one module tuned for precision (Figure 9a), the agent either precisely returns home to the target location or it ends up in an attractor location that is part of a repeating pattern of possible attractors. This shows the periodic nature of the grid cell encoding of space. With one module tuned for range (Figure 9b), all but one of the 500 trials terminate in a cluster centered on the target location. However, the improved range has carried a penalty of worse precision. This penalty is seen to be mostly alleviated by integrating information from multiple grid modules; in Figure 9c, where four grid modules are used, all but six of the 500 trials end up within 0.5 m of the target, with only one ending up more than 1 m away.

To get a sense for the trajectories the model follows during these attempts to reach the target, Figure 10 contains traces of the 50 runs from Figure 9c with the farthest goal distance at the end of the random walk. With some exceptions, the paths taken are all largely straight lines toward the target. All but one trajectory (seen near the top) end up within 0.5 m of the goal.

Discussion

The basis for this project was to use neural representations of space for direction-finding in a robot. The paper has pre-


(a) One grid module, tuned for precision ($g_1 = g_{max}$):

(b) One grid module, tuned for range ($g_1 = g_{min}$):

(c) Four grid modules:

Figure 9: End locations after returning home, for various configurations of grid cell modules. 500 runs for each configuration. Gray dots show end of random walk and black dots show end of return home. Histograms below show the distribution of goal distances at trial termination, with a bin size of 15 cm. (The upper-left run in part b stopped at a distance of 29.7 m from the target.)

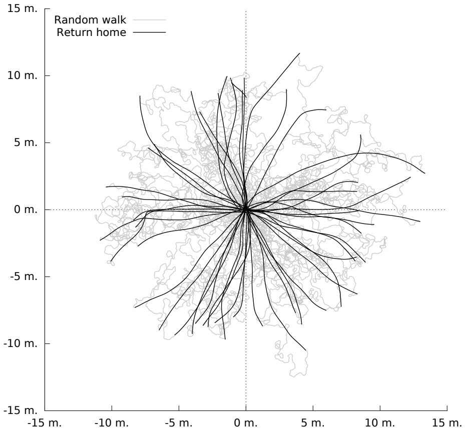


Figure 10: Traces of the 50 runs from Figure 9c with the farthest distance from the goal at the end of the random-walk stage.

sented a model that integrates an existing model of path integrating grid cells with a novel mechanism that is able to use the grid cell representation to direct the agent to a remembered goal. The successful simulation results show promise for using the model in a physical robot in the future. The translation into the physical world will bring with it its own set of challenges, such as noisy self-motion inputs and imprecise motor control. The integration of sensory information into the model is thus one important area for further study, as has been done in other grid cell-based robot controllers (Milford and Wyeth, 2010).

We consider the model at its current abstraction level to be biologically plausible. The inputs and outputs of the model are geocentric direction and speed signals, which is supported by the existence of head-direction cells. Attractor-network models are considered viable candidates for understanding the operation of grid cells, and the phase offset detectors and motor-output neurons are simple input-summing neurons. The target state signals are assumed to be a grid cell-encoding of the target coordinates; this could be provided in the form of backprojections from the hippocampus.

Chadwick et al. (2015) used VR-supported fMRI to look for a goal-direction signal in the human brain. They found that there would be similar brain activity patterns in the entorhinal/subicular region when a given geocentric direction was used as either the current facing direction or the goal direction, and the activity patterns were found to be best accounted for as a mixture of the encodings of the facing direction and the goal direction. The authors see these results as evidence that some form of goal-directed simulation of spatial representations is involved in navigation, citing the model by Erdem and Hasselmo (2012) as an example of how this mechanism could work.

This is not the only possible conclusion of these results, especially since the model by Erdem and Hasselmo requires simulation of look-ahead trajectories in many different directions from the current location in order to discover the goal location. A mechanism similar to the one presented in this paper would allow the goal direction to be calculated directly from grid cell representations of the current location and the goal, avoiding the need for extensive simulations in multiple directions. The mixture representations reported by Chadwick et al. could still conceivably be accounted for by oscillations in the entorhinal/subicular region between encodings of the present and the future spatial states. Experiments at finer spatial and temporal resolutions would hopefully be able to distinguish the extents of these two types of contributions.

Acknowledgements

The author is grateful to Keith Downing and Trygve Solstad for helpful discussions, feedback and advice.

References

Burak, Y. and Fiete, I. R. (2009). Accurate path integration in continuous attractor network models of grid cells. *PLoS computational biology*, 5(2):e1000291.

Chadwick, M. J., Jolly, A. E., Amos, D. P., Hassabis, D., and Spiers, H. J. (2015). A goal direction signal in the human entorhinal/subicular region. *Current Biology*, 25(1):87–92.

Erdem, U. M. and Hasselmo, M. (2012). A goal-directed spatial navigation model using forward trajectory planning based on grid cells. *European Journal of Neuroscience*, 35(6):916–931.

Giocomo, L. M., Moser, M.-B., and Moser, E. I. (2011). Computational models of grid cells. *Neuron*, 71(4):589–603.

Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., and Moser, E. I. (2005). Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436(7052):801–806.

Kubie, J. L. and Fenton, A. A. (2012). Linear look-ahead in conjunctive cells: an entorhinal mechanism for vector-based navigation. *Frontiers in neural circuits*, 6.

McNaughton, B. L., Battaglia, F. P., Jensen, O., Moser, E. I., and Moser, M.-B. (2006). Path integration and the neural basis of the ‘cognitive map’. *Nature Reviews Neuroscience*, 7(8):663–678.

Milford, M. and Schulz, R. (2014). Principles of goal-directed spatial robot navigation in biomimetic models. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1655):20130484.

Milford, M. and Wyeth, G. (2010). Persistent navigation and mapping using a biologically inspired SLAM system. *The International Journal of Robotics Research*, 29(9):1131–1153.

O’Keefe, J. and Dostrovsky, J. (1971). The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 34(1):171–175.

Spiers, H. J. and Barry, C. (2015). Neural systems supporting navigation. *Current Opinion in Behavioral Sciences*, 1:47–55.

Stensola, H., Stensola, T., Solstad, T., Frøland, K., Moser, M.-B., and Moser, E. I. (2012). The entorhinal grid map is discretized. *Nature*, 492(7427):72–78.

An Integrated Neuromechanical Model of Steering in *C. elegans*

Eduardo J. Izquierdo[1] and Randall D. Beer[1,2]

[1]Cognitive Science Program, Indiana University, Bloomington, IN 47406, USA
[2]School of Informatics and Computing, Indiana University, Bloomington, IN 47406, USA
edizquie@indiana.edu

Abstract

In this paper, we extend our previous model circuit for steering in *C. elegans* to control a more realistic biomechanical model of forward locomotion. We show that the identified steering circuit is sufficient to steer the full body during forward locomotion while only innervating a few of the anterior most neck muscles. Analysis of the sensorimotor transformation and phasic stimulation experiments provides evidence that the principles of operation for steering discussed in the model are relevant for steering in the worm. Finally, the integration of the steering circuit in a physical model of the full body allows us to compare more closely the properties of the evolved solutions with those of the worm.

Introduction

One of the grand scientific challenges of this century is to understand how a nervous system controls the behavior of an entire animal. The nematode worm *Caenorhabditis elegans* is a uniquely qualified target for integrated brain-body-environment modeling of a complete animal for three main reasons. First, *C. elegans* exhibits a rich behavioral repertoire, including withdrawal responses, crawling, swimming, a variety of taxes, social feeding, male searching, egg-laying, habituation and associative conditioning (Hart, 2006). Second, a variety of techniques exist for characterizing its behavior and for monitoring and manipulating simultaneously the activity of multiple neurons in the freely moving animal (Shipley et al., 2014). Finally, its "connectome" (302 neurons, 6393 chemical synapses, 890 electrical junctions, and 1410 neuromuscular junctions) is almost completely known (White et al., 1986; Varshney et al., 2011).

The neural circuits involved in specific behaviors in the worm are increasingly being mapped out experimentally (de Bono and Maricq, 2005). Yet, despite the substantial anatomical and neural connectivity knowledge in *C. elegans*, information about the electrophysiological properties of its nervous system is much less complete. For this reason, our approach involves using a real-valued evolutionary algorithm to determine values of the unknown electrophysiological parameters that optimize a behavioral performance measure on the entire system.

Spatial orientation is one of the most fundamental behaviors in *C. elegans*. Recent work has focused on understanding the neural basis of steering during klinotaxis (Iino and Yoshida, 2009; Kato et al., 2014; Luo et al., 2014; Hendricks and Zhang, 2013; Yoshida et al., 2012; Lockery, 2011; Kim et al., 2011). The strategy involves making gradual adjustments to the dorsoventral head swings during forward locomotion, effectively combining information about changes in the environment with information about its body posture to decide in what direction to turn.

In previous work (Izquierdo and Beer, 2013), we identified and modeled the structure of the minimal circuit involved in steering in chemical gradients. The neurons in the identified circuit have since been validated experimentally (McCormick, 2013). Importantly, the same circuit is likely to be involved in steering in other sensory modalities (Kocabas et al., 2012). Our model of steering focused primarily on the neuroanatomical structure and the available neurophysiological properties of the circuit. For simplicity, however, the body and model of movement were highly idealized. Importantly, the model assumed that steering occurred through modulation of the amplitude of the rhythmic oscillations in the head, and that the changes in orientation produced in the head were propagated backwards without explicitly modeling the ventral circuit, muscles, and physical body responsible for locomotion.

In this paper, we extend the steering circuit identified in (Izquierdo and Beer, 2013) to control a more realistic biomechanical model of forward locomotion of the entire worm. Several such models have been proposed in the literature (Gjorgjieva et al., 2014; Cohen and Sanders, 2014). To date, the most complete and biologically-grounded model was proposed by Boyle, Berri and Cohen (2012) (henceforth BBC). The model combines a physical model of the body and the environment with an idealized model of the ventral circuit, as well as neuromuscular control by a sensory feedback mechanism. We use the evolutionary methodology to explore configurations of the steering circuit that can produce klinotaxis in the locomotion model. The resulting integrated neuromechanical model of steering allows us to examine more closely the similarities and differences between the behavior of the model and the worm.

Model and Methods

Model

Environment In the laboratory, spatial orientation is typically studied in petri dishes containing a layer of agar gel (Iino and Yoshida, 2009). Although the BBC forward locomotion model has been demonstrated in the continuum of viscosities ranging between water and agar, our experiments focus exclusively on the latter.

Body We reimplemented the BBC forward locomotion model from published descriptions and publicly available code (Boyle et al., 2012). Using specialized sparse matrix and linear algebra routines, and a semi-implicit backward Euler integrator, we have been able to substantially improve the execution speed of the original implementation on agar, making it feasible for evolutionary optimizability. The worm is modeled in 2D cross-section due to the fact that it normally locomotes on its side, bending only in the dorsal-ventral plane. The $\sim$1mm long continuous body of the worm is divided into variable-width discrete segments (Fig. 1A(i)), each of which are bounded by two cross-sectional rigid rods (black) whose endpoints are connected to their neighbors via damped spring lateral elements (red) modeling the stretch resistance of the cuticle and damped spring diagonal elements (blue) modeling the compression resistance of internal pressure. The rest lengths, spring constants and damping constants of the lateral and diagonal elements are taken directly from BBC, who estimated them from experiments with anesthetized worms. The forces from the lateral and diagonal elements are summed at the endpoints of the rods and then the equations of motion are written for the center of mass of each rod. Since each rod has two translational (x, y) and one rotational (ϕ) degrees of freedom, the body model has a total of 147 degrees of freedom. All kinematic and dynamic parameters are identical to those used by BBC.

Muscles Following BBC, muscles are modeled as damped springs with activation-dependent rest lengths, spring constants and damping constants, endowing them with simplified Hill-like force-length and force-velocity properties. Each discrete lateral element of the body model corresponds to a distinct muscle (red, Fig. 1A(ii)). Since the model is 2D, we combine the bundles DR and DL into a single set of 24 dorsal muscles, each with twice the strength, and likewise for the two ventral bundles. In lieu of experimental constraints on muscle responses, BBC hand-tuned muscle parameters so that realistic-looking locomotion was obtained when the body was coupled to their neural model. Muscle parameters are identical to those used by BBC.

Forward locomotion circuit Forward locomotion is produced by the ventral cord circuit. Following BBC, the model consists of 12 repeating units, each containing one motor neuron of each class: DB, VB, DD, and VD (Fig. 1A(iii)), all of which are known to be necessary for forward locomotion. DD (VD) neurons receive input from VB (DB) motoneurons and inhibit the opposing dorsal (ventral) muscles. In addition, VD also inhibits VB. D-class neurons are modeled as passive (linear) elements. B-class neurons receive input from stretch-receptors. The BBC locomotion circuit model ignores electrical synapses.

Steering circuit The neuroanatomical model used for this study was proposed in our previous study of klinotaxis (Izquierdo and Beer, 2013). The model consists of the minimal circuit (Fig. 1B) connecting the main salt chemosensory class ASE to the neck motor class involved in modulating the amplitude of the sinusoidal locomotion, SMB. The circuit was identified by mining the *C. elegans* connectome and constraining it using existing experimental and theoretical considerations. Chemosensory neurons were modeled after ASE cells. The activation of sensory neurons was modeled as an instantaneous function of a derivative operator $D(t)$ applied to the recent history of attractant concentration. This operator was defined as $D(t) = c_N(t) - c_M(t)$, where $c_N(t)$ is the average concentration over the interval $[t - N, t]$, $c_M(t)$ is the average concentration over the contiguous interval $[t-(N+M), t-N]$, and N and M are the durations of the two intervals. In the case of the OFF cell, the sign of $D(t)$ was inverted so that decreases in concentration yielded positive activations. For both ON cells and OFF cells, negative activations were set to zero. Interneurons and motorneurons were modeled as passive, isopotential nodes,

$$\tau_i \frac{dy_i}{dt} = -y_i + \sum_{j=1}^{N} w_{ji}\sigma(y_j + \theta_j) + \sum_{k=1}^{N} g_{ki}(y_k - y_i) + I_i$$

where y represents the membrane potential (or neuron activation) relative to the resting potential (thus y can assume positive and negative values); τ is the time-constant; θ is a bias term that shifts the range of sensitivity of the output function; w_{ji} represents the strength of the chemical synapse; g_{ki} represents the gap junction conductance between cell k and i ($g_{ki} > 0$). Chemical synapses were modeled as a sigmoidal function of presynaptic voltage, $\sigma(x) = 1/(1 + e^{-x})$. Neck motor neurons included self-connections representing the voltage dependence of inward currents. Unlike the previous model of steering, the circuit does not receive oscillatory component from a rhythmic pattern generator. The output of the motoneurons is fed directly to the anterior-most neck muscles (Fig. 1A(i)).

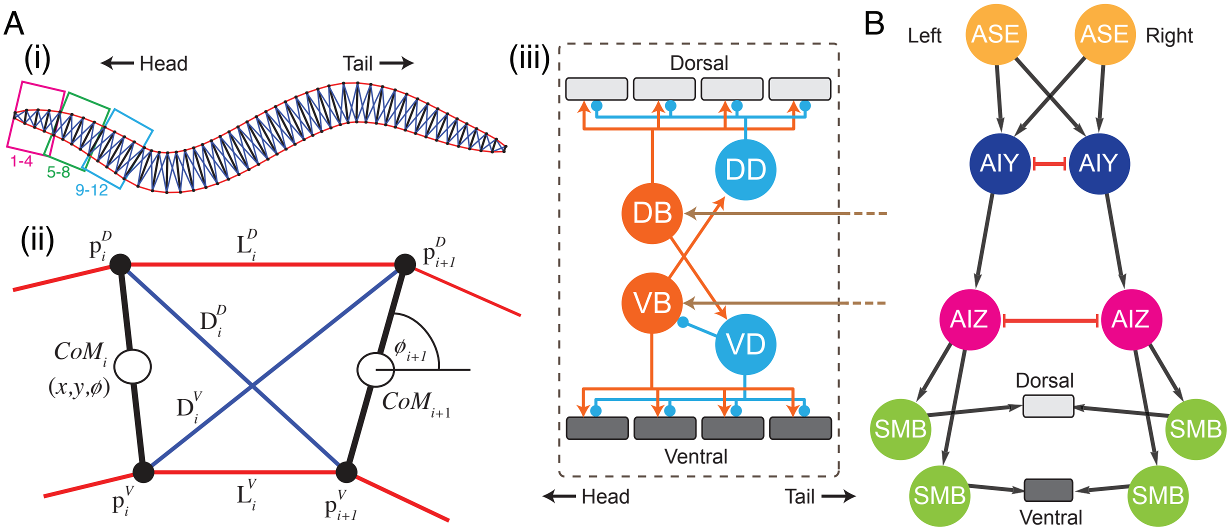


Figure 1: Integrated neuromechanical model. (A) Model of forward locomotion adapted from (Boyle et al., 2012). (i) Complete physical model. The steering circuit affects only the anterior most muscles in the body. We perform evolutionary runs where the steering circuit affects a different set of muscles: 1-4, 1-8, 1-12, and 5-8. (ii) Individual segment in the physical model. (iii) Neuromuscular model. One of 12 repeating units making up the circuit for forward locomotion. The circuit includes a pair of B-class excitatory neurons (green), a pair of D-class inhibitory neurons (light blue), and four muscles (gray) on each side. Synapses are excitatory (arrowhead) or inhibitory (circle head). Posteriorly directed lines from B-class neurons denote the stretch receptor input (brown). (B) Steering neuromuscular model adapted from (Izquierdo and Beer, 2013). Chemosensory class, ASE (orange). Interneuron classes: AIY (blue) and AIZ (magenta). Neck motor neuron class: SMB (red). Dorsal and ventral muscles (gray). All classes have left and right cells. Motor neurons have additional dorsal and ventral pairs of cells. Chemical synapses shown as black arrows. Gap junctions shown as red undirected connections.

Methods

Evolutionary algorithm Although the anatomical connectivity of the nervous system of the nematode worm *C. elegans* has been reconstructed completely (White et al., 1986), the signs and strengths of the anatomical contacts are almost entirely unknown. We used an evolutionary algorithm to explore the space of unknown parameters of the steering circuit such that integrated model produced klinotaxis. We optimized the following parameters of the steering circuit (ranges are shown in brackets): w $[-15, 15]$; θ $[-15, 15]$; g $[0, 2]$; and N and M $[0.1, 4.2]$. Circuit parameters were symmetrical across the dorsal/ventral midline. Parameters were encoded in a 22-element vector of real-values between [-1, 1]; when needed, parameters were linearly mapped to their corresponding ranges. The algorithm was run for populations of 100 individuals. Each time the algorithm was run, individuals were initialized by random selection from the range of each parameter. Populations were evolved for 200 generations. At the end of a run, the parameters of the best performing individual were stored for later analysis.

Fitness function Fitness was evaluated by simulating the integrated model on three trials (Fig. 2). At the start of each trial, the model worm was placed in an environment with a conical gradient with the peak located at different relative angles from the starting orientation of the worm: $-\pi/2, 0, \pi/2$, corresponding to dorsal, straight, and ventral orientations. The peak was located 4.5 cm away from the worm. Model worms were allowed to move for $T = 50$ secs. For each trial, the performance of an individual was measured as the time average of the angle,

$$f = 1 - \frac{1}{T} \int_0^T \frac{a(t)}{\pi} dt \qquad (1)$$

where $a(t)$ is the difference in angle between the gradient peak and the head of the worm at time t. The fitness of an individual was defined as the worst over the three assays. For the conical gradient, attractant concentration $c(t)$ was defined as $c(t) = -\alpha\sqrt{x(t)^2 + y(t)^2}$, where α determines the steepness of the gradient (fixed to 1).

Chemotaxis assay The fitness evaluation is deliberately brief in duration and limited in variation to make the long evolutionary runs feasible computationally. In order to evaluate the ability of the worm to perform chemotaxis more thoroughly, we simulated the worm over more trials (8) of longer durations each (500secs) (see e.g., Fig. 3). In

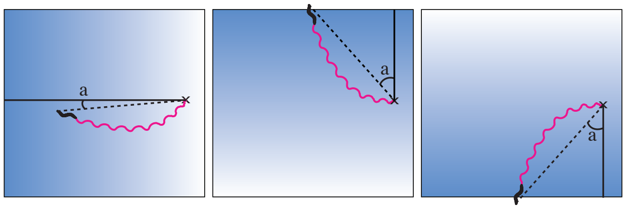


Figure 2: Illustration of fitness evaluation conditions. The gradient in the background represents the chemical concentration (blue). Starting point (x). Worm trace over time (magenta). Difference in angle (a) between the gradient peak (solid line) and the head of the worm at time t (dashed line).

addition to conical gradients, we also tested the worm in more realistic Gaussian gradients: $c(t) = c_0 \exp(-(x(t)^2 + y(t)^2)/2\lambda_c^2)$, with $c_0 = 15$ and $\lambda_c = 1.61$. The fitness score was quantified in terms of a chemotaxis index, defined as the time average of the distance to the peak of the gradient,

$$CI = 1 - \frac{1}{T}\int_0^T \frac{d(t)}{d(0)}dt \tag{2}$$

where $d(t)$ is the Euclidean distance to the peak, $d(0)$ is the model worm's initial distance from the peak (4.5 cm), and T is the total simulated assay time (500 sec).

Results

Modulation of neck and head muscles is sufficient for steering

To identify a neuroanatomically constrained circuit that could steer the *C. elegans* forward locomotion model, we ran 40 evolutionary runs. The steering circuit was allowed to affect only the anterior most muscles in the body: 1-4. The best evolved agents achieved a fitness of 0.78 after 200 generations of evolution. As the circuits were evolved under constrained conditions (see Methods, Fig. 2), we tested the performance of the complete ensemble (i.e., best agent from each evolutionary run) on assays of longer durations that allow them to reach the gradient peak. We measured their performance using the chemotaxis index (see Methods) over 8 trials of 500 secs each, where the initial orientation of the worm was drawn systematically over the range $[0, 2\pi)$ (Fig. 3A). The best performing agent achieved a chemotaxis index of 0.71, with perfect reliability (i.e., the agent reached the peak of the gradient on all trials). This result shows the performance of this agent is comparable to the previous model of klinotaxis (Izquierdo and Beer, 2013). All further analysis was limited to this high-performance individual.

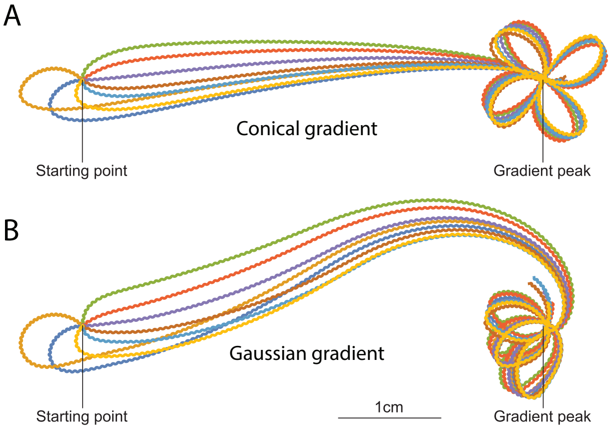


Figure 3: Behavior of the best evolved agent on a conical gradient (A) and a Gaussian gradient (B) over different initial orientations.

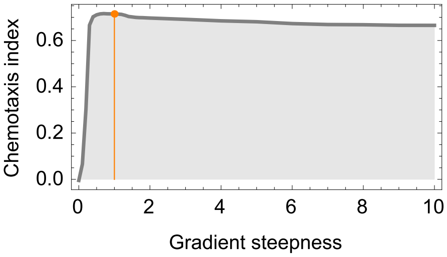


Figure 4: Performance of the best evolved agent in conical gradients of different steepness. Condition employed during evolution in orange ($\alpha = 1$).

To test for generalization we also placed the model worm in a Gaussian-shaped chemical gradient (see Methods), which more closely resembles the gradients used in laboratory tests of chemotaxis in *C. elegans*. The performance in the Gaussian gradient was similar to that obtained in the conical gradient (Fig. 3B). The best agent achieved a chemotaxis index of 0.68, and similarly perfect reliability. The small drop in chemotaxis index reflects the longer path to the peak. Overall, the high performance suggests that the evolved circuit is not specialized for the shape of the gradient; instead, it embodies a more general solution to the task of steering.

Unlike conical gradients, the steepness in Gaussian-shaped gradients varies as a function of the distance to the peak. To understand the worm's ability to generalize, we explored systematically its performance in conical gradients of varying steepness (Fig. 4). The performance is highest for the gradient steepness it was evolved for ($\alpha = 1.0$). If the gradient is too shallow ($\alpha < 0.4$), the performance drops; otherwise, the model worm performs chemotaxis successfully under a wide range of gradient steepnesses, including an order of magnitude larger than what it evolved for. Such robustness allows it to perform well under gradients of different shapes, including the Gaussian gradient.

Ultimately, that successful solutions were found demon-

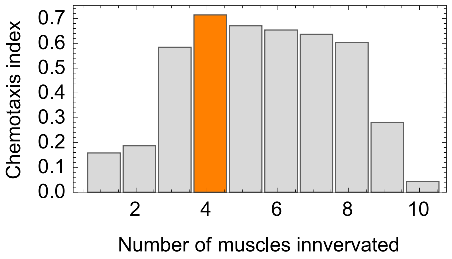


Figure 5: Performance of the best evolved agent when innervating different number of anterior-most muscles. Condition employed during evolution in orange (first 4 muscles).

strates that the identified klinotaxis circuit innervating only the anterior most muscles is sufficient to steer the full body during forward locomotion.

Steering is possible by modulating a wide range of the anterior-most muscles

In our model, the steering circuit innervates the four most anterior muscles (Fig. 1A). However, what muscles are involved in steering in the worm is not yet known (Satoh et al., 2014; Gjorgjieva et al., 2014). We use our approach to explore the range of possibilities for steering within the constraints provided by the forward locomotion model.

First, how does the performance of the best evolved circuit change when the number of muscles innervated by the neck motor neurons varies? We can measure the chemotaxis index of the model as we vary the number of muscles the steering circuit innervates (Fig. 5). Not surprisingly, the best performance is obtained when the circuit is innervating the muscles it evolved to modulate (first four muscles, orange bar). However, the experiment shows that the circuit can perform chemotaxis when modified to innervate anywhere between 3 and 8 of the anterior most muscles. The results suggest the circuit is flexible with respect to the number of muscles necessary to produce chemotaxis.

An analysis of the best evolved agent gives us only a measure of the flexibility that specific solution has to perform chemotaxis under conditions it has never experienced before. But is there an optimal number of muscles that the worm can modulate to produce steering? We can begin to address the second and more general question of optimality by performing additional evolutionary runs on a range of different conditions: when the steering circuit is innervating 4, 8 and 12 of the anterior-most muscles. When we performed 20 evolutionary runs for each of the different conditions, solutions were found that could perform chemotaxis nearly equally well (Fig. 6). This result suggests solutions using a relatively wide range of the anterior-most muscles are equally viable. These experiments, however, assume the anterior most muscles are needed for steering. Is this the case? To address this, we ran an additional evolutionary experiment where the steering circuit could modulate only muscles 5-8. Circuits under this condition consistently failed to evolve a good enough fitness (>0.7). This result suggests modulating anterior-most muscles is necessary for steering. Otherwise the circuit is flexible to perform chemotaxis with a wide range of anterior-most muscles.

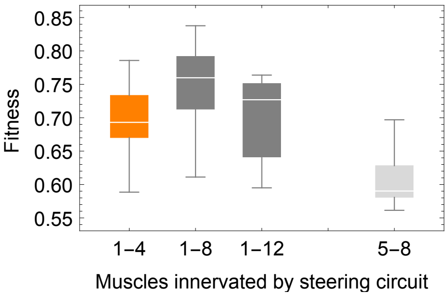


Figure 6: Evolving circuits that innervate a different arrangement of anterior-most muscles. Final fitness of the best individual over 40 evolutionary runs for different conditions.

Sensorimotor transformation is consistent with previous klinotaxis models and experimental observations

In order to understand the sensorimotor transformation that the circuit evolved, we analyzed the orientation responses produced by single stepwise changes in concentration given at different phases of the locomotion (Fig. 7). Orientation responses were expressed in terms of turning bias, computed over several cycles of locomotion following the concentration step. We observed that turning bias varied as a sinusoidal function of the phase of locomotion at which the concentration change occurred (Fig. 7A). Throughout a dorsal head sweep (between phase 0 and π), an increase in concentration resulted in a dorsal reorientation; during a ventral head sweep (between phase π and 2π), an increase in concentration resulted in a ventral reorientation (blue traces). As a response to a decrease in concentration, the worm reorients in the opposite direction to the instantaneous velocity of the head at the time of the step (yellow traces). We also observed that larger changes in concentration led to larger changes in turning (dashed trace vs. solid trace). The results were consistent with our previous models of steering in an idealized body (Izquierdo and Beer, 2013).

In order to illustrate how this sensorimotor transformation can lead to successful klinotaxis, we show example traces without changes in concentration (black trace) and with positive and negative changes in concentration (blue and yellow, respectively) given at the beginning of a dorsal (phase 0) and a ventral (phase π) head sweep (Fig. 7B). The instantaneous velocity vector of the head at the time of an upstep signals the direction of the peak implied by such a step (blue dashed arrow). As a result of a positive change in concentration, the worm moves in the direction of the implied peak (blue traces). A downstep in concentration given at that same phase in the locomotion leads to turning in the opposite direction of the implied peak (yellow traces). Thus, the model worm appropriately corrects its orientation relative to discrepancies between its velocity vector and the direction of the peak.

A full sensorimotor transformation analysis has not yet been performed in the worm. However, recent experiments have analyzed the movement of the worm during phasic stimulation to the sensory neurons (Kocabas et al., 2012; Satoh et al., 2014). We replicated these experiments in the artificial worm (Fig. 7C). When the ON cell is stimulated repeatedly during dorsal head sweeps, the worm displays a dorsal-oriented curvature (green trace). When the ON cell is stimulated repeatedly during ventral head sweeps, the worm displays a ventral-oriented curvature (blue trace). The curvatures are different for dorsal and ventral head-sweeps due to a dorsoventral asymmetry in the BBC locomotion model. The effect is the opposite when stimulating the OFF cell (not shown). The results of these experiments follow directly from the sensorimotor transformation provided in Fig. 7A, and crucially, they reproduce the results observed in the worm. These results suggest that the model and biological circuit may be operating according to similar principles. Furthermore, they predict the overall sensorimotor transformation in the worm should be consistent with the integrated model and previous theoretical models (Fig. 7A).

Shape statistics analysis reveals eigenworms dedicated to steering

The integration of a steering circuit in a biomechanical model of the body allows us to compare more closely the properties of the evolved solutions with those of the worm. Several approaches have been suggested for mathematically describing locomotion in *C. elegans*. One of the most useful approaches involves using video microscopy of the worm's movement to find a low dimensional description of the macroscopic behavior (Stephens et al., 2008). Analysis of behavioral data has shown that the space of shapes adopted by *C. elegans* can be almost completely described (95% of the variance) by projections along four principal "eigenworms" (Stephens et al., 2008). Their analysis also suggests the first two eigenworms are sinuous and together offer a quantitative characterization of the traveling wave along the body during forward locomotion.

In order to analyze the behavior of the model and compare it to that of the worm, we use detailed data of the simulated body over time during freely moving behavior. We proceeded in two steps. First, as results of the shape statistics have not been replicated in a simulated model of the worm yet, we analyzed the forward locomotion model without the steering circuit and compared the results with those of the worm. As a second step, as an analysis of the shape

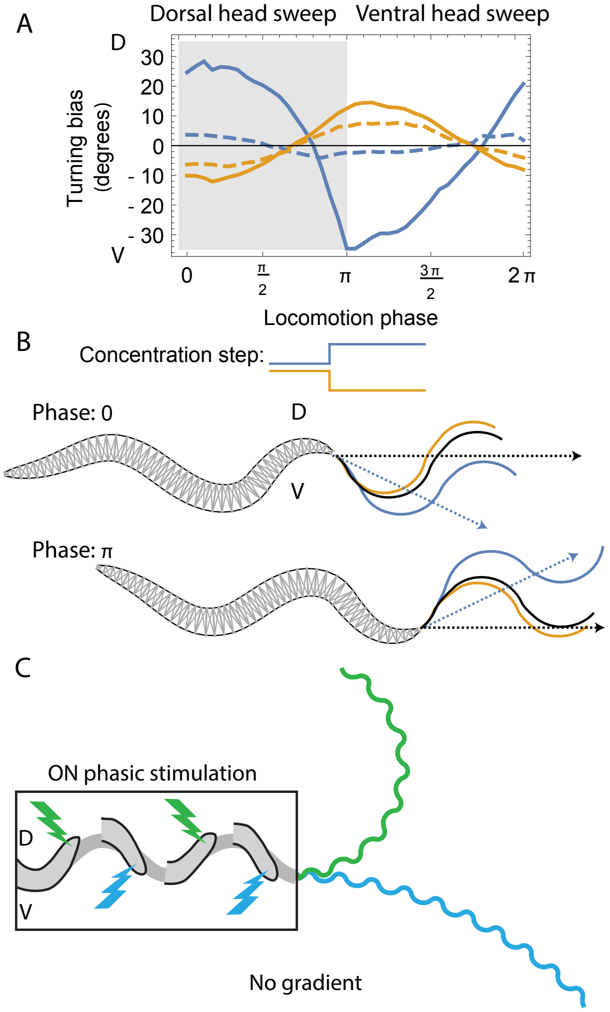


Figure 7: Sensorimotor transformation. (A) Phase sensitivity of orientation responses. Response to upsteps (blue). Response to downsteps (yellow). Plots show turning bias versus the phase of locomotion at which the concentration step occurred. Dashed traces indicate the response during a step of smaller magnitude. Dorsal head sweep marked in shaded region. (B) Shape of the body during different phases of locomotion and the effect of positive and negative changes in concentration on the translational direction of the worm. Increase/decrease in concentration at the start of a ventral head sweep (phase 0) leads to a dorsal/ventral bias in the translational direction of the worm (blue/yellow trace) relative to the unstimulated condition (black trace). The opposite is true at the start of a dorsal head sweep (phase π). In all cases, the model worm corrects its orientation towards the peak implied by the change in concentration. (C) Phasic stimulation leads to worm track curvature in the absence of gradients. Stimulation to the ON cell during ventral head-sweeps (phase $0 - \pi$) leads to ventral curvature (blue trace). Stimulation to the ON cell during dorsal head-sweeps (phase $\pi - 2\pi$) leads to dorsal curvature (green trace).

statistics has not been performed in the context of steering behavior exclusively, we analyzed the integrated model during steering as a way to predict the resulting shape statistics in the worm.

Without the steering circuit, the forward locomotion model displays a covariance matrix of fluctuations in angle that is rather similar to that of the worm (Fig. 8A): (a) Inhomogeneity along the diagonal shows motion is not sinusoidal; (b) The smooth structure of the matrix suggest only a small number of modes are significant. The main difference between the covariance matrices of model and worm is towards the posterior, which seems to be flatter in the model than in the worm (cf. Fig. 2A in Stephens et al. (2008)). This suggests the BBC model of forward locomotion may be missing crucial aspects of the operation of the circuit that allow for the wave to propagate backwards more fully. When we consider the eigenvalues of the model, the majority of the variance (98.7%) is captured by the first two eigenvalues (red, Fig. 8B). This is expected because the model without the steering circuit is only moving forwards; there are no gradual turns, foraging, or omega turns. Associated with each dominant mode is an eigenworm. The first two eigenworms of the model are also similar to those of the worm (red, Fig. 8C). Indeed, these two eigenworms capture all the variability along the body (Fig. 8D). This result is consistent with analysis of forward locomotion in the worm (Stephens et al., 2008).

When the steering circuit is added to the model worm, the covariance matrix of fluctuations in angle changes significantly (Fig. 8A). Although there is still inhomogeneity along the diagonal, the structure of the matrix is not as smooth, suggesting it takes more modes to capture the variance of the movement. Indeed, while the first two eigenvalues still capture the majority of the variance (59.1%), each additional eigenvalue is responsible for only a small amount of the additional variance (blue, Fig. 8B). As with the worm and the model without steering, the first two eigenworms correspond to those responsible for forward locomotion (blue, Fig. 8C). Interestingly, the third eigenworm reduces information mostly in the neck and also in the tail (green area, Fig. 8D). This suggests the possibility of an eigenworm dedicated mainly to steering. However, together these three modes only capture 64.7% of the variability in shapes adopted by the worm during steering. These results prompt us to analyze the shape statistics during steering behavior exclusively in the worm. Differences between the predicted shape statistics in the model and the worm is likely to suggests ways in which the model of the body does not propagate the steering in the same way as in the worm.

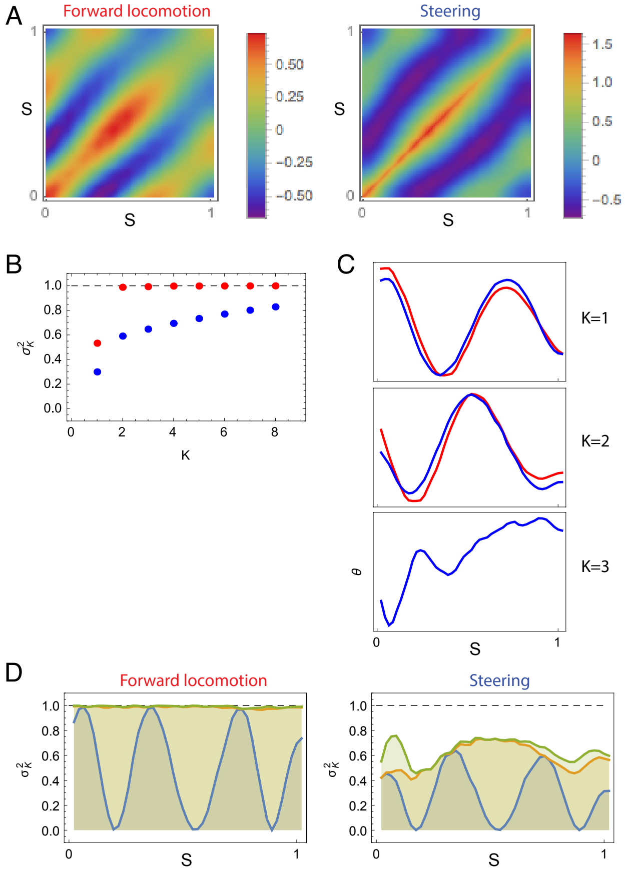


Figure 8: Shape statistics: Eigenworms for forward locomotion and steering. Position along the body is represented by s, normalized so that $s = 0$ is the head and $s = 1$ is the tail. (A) The covariance matrix of fluctuations in angle. (B) Fraction of the total variance (integrated along the body of the worm) captured by keeping K eigenvectors ($K = 1$ to 8), calculated from the Eigenvalues of the covariance matrices for the forward locomotion model (red) and the integrated model with the steering circuit (blue). (C) Associated with each dominant mode is an eigenvector or eigenworm. First three eigenworms shown for the forward locomotion model (red) and the integrated model with the steering circuit (blue). (D) Fraction of the variance (unrolled over the body of the worm) captured by keeping K eigenvectors ($K = 1$ to 3, from bottom to top) for the forward locomotion model and the integrated model with the steering circuit.

Conclusion

In this paper, we extended our previous model circuit for steering in *C. elegans* to control a more realistic biomechanical model of forward locomotion. The evolutionary algorithm successfully found several combinations of parameters of the steering circuit capable of performing klinotaxis. The existence of solutions suggests that the identified steering circuit, while only innervating a few of the anterior most muscles, is indeed sufficient to steer the full body during forward locomotion. Furthermore, analysis of the sensorimotor transformation showed that the best evolved circuit is both:

(a) qualitatively similar to previous theoretical models of steering (Izquierdo and Lockery, 2010; Izquierdo and Beer, 2013), and (b) consistent with recent experiments involving phasic stimulation of sensory neurons in the worm (Kocabas et al., 2012; Satoh et al., 2014). This suggests that the principles of operation for steering discussed in the idealized models are relevant for steering in the nematode. Finally, the integration of the circuits in a biomechanical model of the full body allowed us to compare more closely the properties of the evolved solutions with those of the worm. Analysis of the shapes produced by the steering model revealed the first two eigenworms are responsible for forward locomotion and the third is likely to be involved in steering. Each of these findings corresponds to an experimental prediction that could potentially be tested in the worm.

There are several directions for future work that we believe will be productive. First, as new experimental evidence becomes available, the models have to be revisited, updated, and expanded. Since nearly the entire behavioral repertoire of *C. elegans* is ultimately expressed through movement, this is particularly crucial for the neuromechanical basis of locomotion and its modulation – the foundation upon which analyses of all other behaviors must build. Second, as neural circuits involved in different behaviors in the worm continue to be mapped out experimentally, parallel efforts to develop embodied and situated models of these circuits is likely to accelerate the understanding of the neural basis of their behavior. In addition to modeling each of the distinct behaviors individually, future work will focus on integrating these multiple circuits, responsible for different behaviors, under the same body. Ultimately, incrementally constructing a comprehensive model of a complete organism is likely to completely transform our understanding of how integrated behavior arises from the ongoing interaction of an animal's nervous system, its body, and its environment.

Acknowledgements

This work was supported in part by NSF grant IIC-1216739.

References

Boyle, J. H., Berri, S., and Cohen, N. (2012). Gait modulation in *C. elegans*: An integrated neuromechanical model. *Frontiers in Computational Neuroscience*, 6:10.

Cohen, N. and Sanders, T. (2014). Nematode locomotion: dissecting the neuronal–environmental loop. *Current Opinion in Neurobiology*, 25:99–106.

de Bono, M. and Maricq, A. V. (2005). Neuronal substrates of complex behaviors in *C. elegans*. *Annual Review of Neuroscience*, 28(1):451–501.

Gjorgjieva, J., Biron, D., and Haspel, G. (2014). Neurobiology of *Caenorhabditis elegans* Locomotion: Where Do We Stand? *BioScience*, 64(6):476–486.

Hart, A. C. (2006). Behavior. In *WormBook*. The *C. elegans* Research Community.

Hendricks, M. and Zhang, Y. (2013). Complex RIA calcium dynamics and its function in navigational behavior. *Worm*, 2(3):e25546.

Iino, Y. and Yoshida, K. (2009). Parallel use of two behavioral mechanisms for chemotaxis in *Caenorhabditis elegans*. *Journal of Neuroscience*, 29(17):5370–5380.

Izquierdo, E. J. and Beer, R. D. (2013). Connecting a connectome to behavior: an ensemble of neuroanatomical models of *C. elegans* klinotaxis. *PLoS Computational Biology*, 9(2):e1002890.

Izquierdo, E. J. and Lockery, S. R. (2010). Evolution and analysis of minimal neural circuits for klinotaxis in *Caenorhabditis elegans*. *Journal of Neuroscience*, 30(39):12908–12917.

Kato, S., Xu, Y., Cho, C. E., Abbott, L. F., and Bargmann, C. I. (2014). Temporal responses of *C. elegans* chemosensory neurons are preserved in behavioral dynamics. *Neuron*, 81(3):616–628.

Kim, D., Park, S., and Mahadevan, L. (2011). The shallow turn of a worm. *Journal of Experimental Biology*, 214:1554–1559.

Kocabas, A., Shen, C.-H., Guo, Z. V., and Ramanathan, S. (2012). Controlling interneuron activity in *Caenorhabditis elegans* to evoke chemotactic behaviour. *Nature*, 490(7419):273–277.

Lockery, S. R. (2011). The computational worm: spatial orientation and its neuronal basis in *C. elegans*. *Current Opinion in Neurobiology*, 21(5):782–790.

Luo, L., Wen, Q., Ren, J., and Hendricks, M. (2014). Dynamic encoding of perception, memory, and movement in a *C. elegans* chemotaxis circuit. *Neuron*, 82(5):1115–1128.

McCormick, K. (2013). *Circuit and behavioral analysis of klinotaxis in Caenorhabditis elegans.* Ph.D. Thesis, University of Oregon.

Satoh, Y., Sato, H., Kunitomo, H., Fei, X., Hashimoto, K., and Iino, Y. (2014). Regulation of experience-dependent bidirectional chemotaxis by a neural circuit switch in *Caenorhabditis elegans*. *Journal of Neuroscience*, 34(47):15631–15637.

Shipley, F. B., Clark, C. M., Alkema, M. J., and Leifer, A. M. (2014). Simultaneous optogenetic manipulation and calcium imaging in freely moving *C. elegans*. *Frontiers in Neural Circuits*, 8:28.

Stephens, G. J., Johnson-Kerner, B., Bialek, W., and Ryu, W. S. (2008). Dimensionality and dynamics in the behavior of *C. elegans*. *PLoS Computational Biology*, 4(4):e1000028.

Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., and Chklovskii, D. B. (2011). Structural properties of the *Caenorhabditis elegans* neuronal network. *PLoS Computational Biology*, 7(2):e1001066.

White, J. G., Southgate, E., Thomson, J. N., and Brenner, S. (1986). The Structure of the Nervous System of the Nematode *Caenorhabditis elegans*. *Phil. Trans. R. Soc. B*, 314(1165):1–340.

Yoshida, K., Hirotsu, T., Tagawa, T., Oda, S., Wakabayashi, T., Iino, Y., and Ishihara, T. (2012). Odour concentration-dependent olfactory preference change in *C. elegans*. *Nature Communications*, 3:739.

CriPS: Critical Particle Swarm Optimisation

Adam Erskine[1] and J. Michael Herrmann[1]

[1]The University of Edinburgh, School of Informatics, 10 Crichton St, Edinburgh EH8 9AB, U.K.
a.erskine@sms.ed.ac.uk

Abstract

Particle Swarm Optimisation (PSO) is a metaheuristic used to solve search tasks and is inspired by the flocking behaviour of birds. Traditionally careful tuning of parameters are required to avoid stagnation. Many animals forage using search strategies that show power law distributions in their motions in the form of Lévy flight random walks. It might be expected that when exploring spaces for optima in the absence of any prior knowledge a similar strategy may be useful. Using feedback from swarm metrics, we engineer modifications to the standard PSO algorithm that induce criticality. Such dynamics show long tail distributions in system event sizes. The presence of large (though few) exploratory steps removes the risk of stagnation. The Critical Particle Swarm (CriPS) can be easily combined with many existing PSO extensions.

Introduction

Nature abounds with swarms: herds of mammals, schools of fish, flocks of birds etc. They demonstrate amazing coordinated movements, often showing near simultaneous direction changes across the whole swarm in the response to an approaching threat. Such a loss of length scales is reminiscent of state changes in systems at the point of criticality. For example the Ising model of ferromagnetic materials shows that small magnetic perturbations to the system can generate system wide responses. Particle swarm optimisation (PSO) is a metaheuristic method for obtaining solutions to optimisation problems (Kennedy and Eberhart, 1995). Inspired by the cooperative behaviours of flocks of birds or schools of fish, it employs a number of particles as a swarm of potential solutions. They share knowledge of the problem space in order to improve the performance of the swarm. The position and velocity of particles are iteratively updated using the following rules (Kennedy and Eberhart, 1995)

$$\mathbf{v}_i' = \omega \mathbf{v}_i + \alpha_2 \mathbf{R}_1(\mathbf{p}_i - \mathbf{x}_i) + \alpha_2 \mathbf{R}_2(\mathbf{g} - \mathbf{x}_i) \qquad (1)$$

$$\mathbf{x}_i' = \mathbf{x}_i + \mathbf{v}_i' \qquad (2)$$

Here $\mathbf{x}_i$ and $\mathbf{v}_i$ represent the position and velocity vectors of the i^{th} particle in the swarm. The velocity update, $\mathbf{v}_i'$, is a linear combination of three contributions: An inertial term parameterised by ω, a pull towards the personal best location $\mathbf{p}_i$ parameterised by α_1, and a pull towards the global best location $\mathbf{g}$ parameterised by α_2. The symbols $\mathbf{R}_1$ and $\mathbf{R}_2$ denote diagonal matrices whose non-zero entries are uniformly distributed in the unit interval.

In order to perform optimally, the algorithm has to ensure an appropriate mix of local search, i.e. the exploitation of existing knowledge of the problem space, and exploration of areas not yet adequately sampled. The optimal balance of exploration and exploitation will depend on the nature of the problem space. In PSO this mix is achieved via the parameterisation of the algorithm. As metaheuristic algorithms are often applied to problems with little formal specification, trial-and-error search remains the only general option for parameter tuning. Inappropriately set parameters frequently lead to poor results with the swarm prematurely converging to a local minima. The particles are said to stagnate.

The CriPS algorithm ensures that a continuing mix of swarm behaviours occurs throughout its execution. The algorithm includes its own parameters that may be used to adjust its online behaviour, however, it is shown that using values derived from properties of the problem space leads to reasonable results. As CriPS uses the standard PSO update rules to locate better solutions its results do not exceed those of modern metaheuristic methods, but the mechanism should be easily combined with such methods.

Particle Swarm Optimisation

To circumvent PSO shortcomings a number of mechanisms for parameter adaption in PSO have been proposed. Various velocity controls have been explored: simple truncation (Kennedy, 1998); or the application of a constriction factor derived from the algorithm's parameters Clerc and Kennedy (2002). Parameter controls may be applied for instance via linear decreases in the ω value as the algorithm progresses (Shi and Eberhart, 1999).

Larger ω values tend to encourage greater exploration by the swarm of the problem space. By reducing this parameter's value the swarm is guided to spend more time exploiting the locations that seem promising. Other approaches to varying PSO's parameters have been explored including, random, increasing, decreasing and chaotic (Bansal et al., 2011). Stagnation tends to still present as a problem. So explicit means to increase diversity have been employed: by using particle repulsion (Riget and Vesterstrøm, 2002; Chowdhury et al., 2013); or random velocities (García-Villoria and Pastor, 2009). The original PSO swarm allowed all particles to share knowledge of global best values. Essentially the swarm was fully connected. Other topologies have been shown to result in performance improvements (Kennedy, 1998; Bratton and Kennedy, 2007).

PSO's original bird flocking inspiration has been joined by many other bio-inspired approaches. The different search mechanisms of different species bring novel exploration and exploitation methods to the algorithm. One recent approach, Cuckoo search (Yang and Deb, 2009), co-opts an earlier idea of using the power-law distributed movements of Lévy flights to drive local searches.

Algorithms may use an adaptive probability model to choose from a number of strategies in a success-dependent fashion. Self-adaptive algorithms have been proposed for differential evolution (Qin and Suganthan, 2005) and also for PSO (Wang et al., 2011; Zhan et al., 2009). These hyperheuristic methods measure the behaviour of the swarm and use this to determine which algorithm is most appropriate to run at that particular time.

The TRIBES mechanism removes the velocity update component of PSO and the inherent need to set parameter values (Clerc, 2010). Multiple sub-swarms grow and shrink (removing the need to set a swarm size) depending on performance. The swarms exchange information on good locations and new positions are created by weighted sums of the positions of particles representing good solutions.

Recent successful approaches have been derived from the Covariance Matrix Analysis Evolutionary Strategy (CMA-ES). This is a (μ, λ) evolutionary strategy. A multivariate normal distribution derived from the μ best parents are used to generate the next generation of solutions (λ in number). After each iteration the distribution's covariance matrix is adapted to direct the future selection toward new and better solutions. The technique may still suffer from premature convergence. However, detection of such stagnation may be used as a signal to trigger algorithmic restarts or other procedures to increase the diversity of the swarm to obtain good results (Loshchilov, 2013).

Criticality

Criticality in equilibrium thermodynamics is used to refer to the properties of a system at a transition point between phases. At this point small perturbations can propagate throughout the whole system (Jensen, 1998). Ferromagnetic materials, for example, gain their magnetism ultimately via the alignment of magnetic dipole moments arising from electron spin states. The state of any individual dipole may be influenced by the magnetic field it finds itself in. If this material is at a high temperature, then the thermal noise will be greater than the influence of small perturbations and no effect is seen. At low temperatures the dipole moments are effectively frozen in whichever state they find themselves in. Small external magnetic perturbations will have little or no effect in either case. When the system is tuned by heating it to point between these two states, then any small external magnetic perturbations are able to flip the state of nearby dipoles. Further, these changes can propagate throughout the material resulting in a chain of dipole flips. A characteristic of the such systems is that changes in system states can exhibit power law distributions.

More widely the term criticality may refer to any dynamical system which behaves in a manner like this (Bak et al., 1988). Bak notes that there are many systems in nature that exhibit similar power law type distributions. We see such heavy tail distributions in the earthquakes described by Gutenburg-Richter law. Similarly power laws crop up in: Zipfs law in ranked distributions of word usage in English; fractal geometries in nature; and 1/f type noise in physical systems. The ferromagnetic example above requires tuning but it seems infeasible for all the occurrences of power-laws to require such fine tuning. It is proposed that systems consisting of many interacting units may evolve automatically into a critical state: effectively self-organising to a point poised between order and chaos (Bak, 1997).

The properties of such systems may be explored by looking at simple models such as the proposed sandpile model (Bak et al., 1988). Sand trickled onto a pile results in gradients that are at some critical angle. Adding a single extra grain may release an avalanche whose size is shown to follow a power law distribution. Similar dynamics have been noted earthquake magnitudes (Olami et al., 1992), punctuated evolution (Bak and Sneppen, 1993), neuronal avalanches (Eurich et al., 2002; Beggs and Plenz, 2003; Levina et al., 2007). The presence of criticality in a system can be shown to optimise the system in some way. Shew et al. (2011) showed the criticality of cortical neuronal avalanches results in optimal information capacity and transmission.

In such systems, the presence of power laws in system event sizes imply that events of any size are possible. Practical limits arise from the finite size of systems. If we were to engineer critical dynamics into the swarm size of a PSO algorithm then we would be assured that the swarm would sooner or later extend throughout the whole problem space and thus avoid stagnation. A random search strategy would likewise avoid stagnation, but our approach assures that the algorithm can still favour exploitation of the problem space over exploration.

Previously there have been approaches to adding criticality to PSO. A sandpile-like approach was employed by adding an additional counter to each particle (Lovbjerg and Krink, 2002). If a particle came close to another it incremented its counter. Once over a threshold the particle relocated within the problem space and redistributed its accrued value to other particles allowing *avalanches* to occur. There was limited evidence that the swarm behaved in a critical manner and the problem of setting parameter values remained. Richer and Blackwell (2006) implemented a PSO algorithm inspired by the Lévy flight random walk behaviour of many foraging animals. They modified a Gaussian PSO algorithm, which performs velocity updates by drawing from Gaussian distributions scaled by distance of particle from local and global best locations, to use Lévy distributions instead. The nature of this power-law approach produced a greater number of outliers in a given problem space, resulting in a more powerfully exploring swarm. Their results showed that this Lévy swarm outperformed both standard PSO and Gaussian PSO approaches. More recently an approach was made where numbers drawn from a power-law distribution were used to modify the PSO parameters directly on each iteration (Fernandes et al., 2012). Again, improvements over standard PSO were apparent.

Our CriPS algorithm aims to remove the need to set parameters. Our approach is to make the algorithm responsive to its environment so that it is able to self-tune to the current problem being explored. The technique presented here was first explored by Cordero (2012). A measure of the diversity of the particle swarm is made. This acts as a feedback signal modifying the PSO algorithm parameters on each iteration.

The CriPS Algorithm

We modify the swarm dynamics of the PSO algorithm such that exploration and exploitation of the problem space are statistically balanced automatically and on-line during the optimisation process. Intuitively, we induce a more stable behaviour should the swarm tend to diverge, but change the parameters of the algorithm towards the unstable regime if the swarm is likely to collapse. In this way we hope to achieve an optimal compromise between local fine-grained search near candidate optima and large scale exploration. The swarm will stay near a critical regime between stability and instability.

Our algorithm uses the dynamic of the swarm itself as a signal to modify its future behaviour. In order to maintain the responsiveness of the swarm to the objective function, we will control the parameters based on changes in the swarm's diversity. The diversity is most obviously a measure of the swarm size. The average distance between every particle and the centroid of the swarm, for instance, could be calculated on successive iterations and this change used as a feedback signal. We could similarly use intra-particle distances as our diversity measure. A third possibility is to measure the average velocity norm of all particles. This measures the dynamics of the swarm in a slightly different way. The speed of a particle is in essence telling us about the ability of the particles to move around in the problem space. Higher speeds represent more exploratory particles. Averaging over all particles gives us a measure of the swarms diversity. The difference of this metric between two successive iterations quantifies the exploratory and exploitive behaviour of the swarm. If the difference is positive, the swarm is accelerating and increasing its tendency to explore, if the difference is negative, the swarm explores less, exploits more. This third metric is used as our measure of swarm diversity.

We use the change in the metric value between iterations to provide a feedback signal to update the parameter values. For any metric S as listed above, the change in its value is

$$\Delta S = S(t+1) - S(t). \tag{3}$$

We update the parameters of the PSO algorithm using

$$\theta(t+1) = \theta(t) - \varepsilon\, f(\Delta S), \tag{4}$$

where θ is each of the PSO parameters: ω, α_1, or α_2. All parameters are updated on each iteration. Alternatively one may update a single stochastically chosen parameter on each iteration. This variant allows exploitation of the full PSO parameter space and appears to further improve results. We wish our updates to be approximately the same order of magnitude as PSO parameters (typically a little smaller). We therefore choose a function $f(\Delta S)$ to provide our update signal within the range [-1, 1]. We used a sigmoid function given in equation 5. The σ parameter adjust the sensitivity of the sigmoid response to the size of changes in the metric. A small value will make the response function tend to return larger update values for smaller changes in the swarm metric used. The ε parameter is used to scale the maximum size of the update.

The rescaled sigmoid, we use here, is given by

$$f(\Delta S) = \tanh\left(\frac{\Delta S}{2\sigma}\right) \tag{5}$$

For our experiments we used the mean velocity norm of the particles as our metric S, our σ is chosen to be less than or equal to the maximum extent of the problem space so that the sigmoid returns either -1 or 1 for swarms where the change in the mean velocity norm exceeds the extent of the problem space per iteration.

The change in the metric provides a negative feedback signal to the PSO parameters. Thus an expanding swarm results in smaller ω, α_1 and α_2 values. Conversely a shrinking swarm leads to increased values. We can consider how this may affect the dynamic of our swarm. A large quickly expanding swarm is exploring the problem space. To encourage exploitation it needs to be brought under control. By decreasing the parameters we will reduce the inertial term ($\omega\mathbf{v}_i$) of the update equation. This will have the effect of reducing the expansion if $\mathbf{v}_i$ is outward. If inward then this component is still reduced, yet it will still contribute to the swarm's shrinkage. The α parameters are also reduced, however their contribution are multiplied by the large distance between the particle and the known good locations already found. The *pull* of these terms is therefore large. In the opposite case, that of a small relatively stagnant swarm, the α terms are multiplied by small distances and contribute little to changing the particles' dynamics. Increasing ω over several iterations will accelerate the particles in some direction, ultimately outward. This requires ω to increase above unity. In this way the feedback mechanism in equation 4 allows the swarm to utilise the changing swarm metric as a feedback signal to control the balance of exploration and exploitation of the problem space. CriPS is shown in Algorithm 1.

```
Input: Objective function F(), Maximum iterations I,
       Target Fitness value V, N particles with position
       and velocity state $x_j$, $v_j$ where $j \in [1, N]$
Output: Returns location of best evaluation of F()
        achieved in I iterations
Initialise:
minSize ← lower limit of F()'s spatial bounds;
maxSize ← upper limit of F()'s spatial bounds;
$x_j \leftarrow$ uniform random [minSize, maxSize];
$v_j \leftarrow$ uniform random [minSize, maxSize];
$\omega \leftarrow 0.815$;
$\alpha_1 \leftarrow 1$;
$\alpha_2 \leftarrow 1$;
$\varepsilon \leftarrow 1$;
$\sigma \leftarrow$ maxSize - minSize;
S ← mean velocity norm;
while Termination conditions are not met do
    Update PSO swarm velocities;
    Update PSO swarm positions;
    Calculate F($x_j$) $\forall j$;
    Update personal and global bests;
    S' ← mean velocity norm;
    $\Delta S \leftarrow$ S' - S;
    foreach PSO parameter $\Theta$ do
        $\Delta\Theta \leftarrow \varepsilon \times \tanh\left(\frac{\Delta S}{2\sigma}\right)$;
        $\Theta \leftarrow \Theta - \Delta(\Theta)$;
    end
    S ← S';
end
```

Algorithm 1: CriPS algorithm.

Many PSO variants restrict their particles to remain within the spatial extent that the test objective function is defined over. We do not wish to constrain our swarm's motion. Instead for locations outside the problem space the objective function returns a maximum real value. In this way the particles can explore all space, but will never locate better locations outside the defined benchmark's spatial extent. The attraction of local and global best locations within the defined problem space always draws the particles back.

Not shown in this algorithm are two components: these apply exponential forces to the parameters should the swarm tend to zero size or a size much greater (fifty times) the size of the problem space. These are required for reasons of practicality. As we apply no constraints to the swarm size a large swarm size change can expand well beyond the problem space. Whilst the dynamics of our update mechanism would sooner or later return the swarm to within the confines of the defined problem, there is the potential to waste time and/or function evaluations. A similar situation may apply for very small swarms. In reality neither *force* appears to be much required.

Results

The parameters ε and σ in equations 4 and 5 scale the size of the PSO parameter updates. First we look at the effect these have on the swarm dynamics. As our aim is to remove the need to set any parameters we will fix the values of all parameters before we explore the evidence of criticality in these swarm dynamics. Next we show that stagnation is avoided. Finally by comparison with modern algorithms and recent benchmark tests we assess the performance of CriPS. Dynamics are explored using simple but non-trivial functions in two dimensions, e.g. Schwefel and Griewank functions. We present typical examples here. The benchmarks tests are detailed later use a wider range of problem functions. For our PSO implementation we choose $\omega = 0.815$, $\alpha_1 = 1$ and $\alpha_2 = 1$. These are also used as the initial values for our CriPS algorithm. There may be better values that could be chosen for PSO, but we are interested here in comparing the behaviours between the two approaches. Performance comparisons are made later with algorithms that competed in the CEC2013 metaheuristic benchmarking competition.

Swarm Dynamics We measure the swarm size by calculating the mean distance of all particles from the swarm centroid (MSD). A typical evolution of this measure is shown in figure 1. For standard PSO it is typical that the swarm gradually shrinks as the algorithm runs, homing in on a preferred solution. This arises from the tendency of particles to become trapped in local minima within the problem space. Whether, or how quickly, this occurs depends on the choice of parameter values. It is not possible to know in advance for an arbitrary function which values will yield the best results. CriPS, in contrast, shows a swarm dynamic that features periods where the swarm is relatively small, with occasional large bursts. These include bursts larger than the problem space explored. It is from this dynamic that stagnation is avoided.

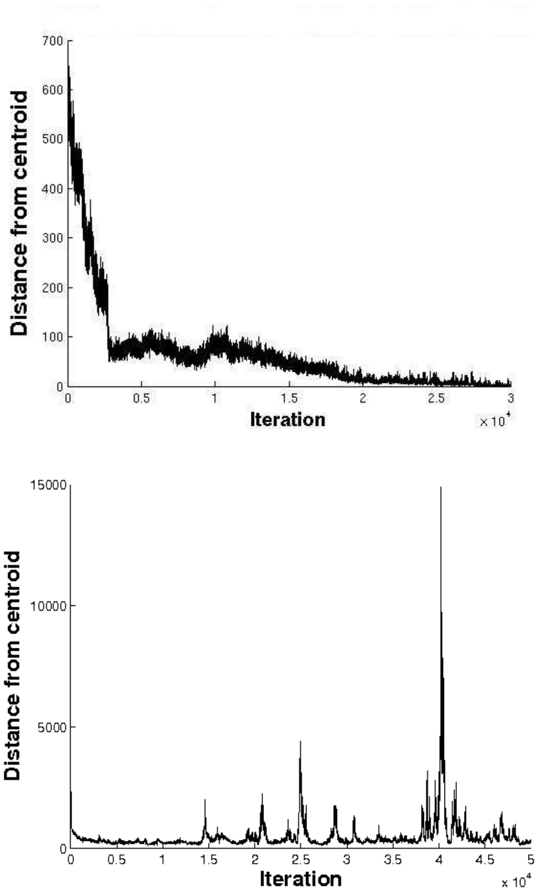


Figure 1: Swarm size dynamic whilst algorithm runs against a two dimensional Schwefel function. The mean swarm distance from centroid (MSD) is plotted as a function of iteration number. Upper: Standard PSO, Lower: CriPS. We used parameter values $\omega = 0.815$, $\alpha_1 = 1$ and $\alpha_2 = 1$ for each.

Assessment of criticality Investigating the presence of power-laws requires the study of larger PSO systems. We increased the number of particles in the swarm to 250 and studied the dynamics for 50000 iterations. We plot the distribution of changes in swarm size between iterations. Our diversity is measured by the mean swarm distance from the swarm centroid (MSD). We can plot the frequency of changes in this (ΔMSD). Figure 2 shows this on a log-log plot. The bottom graph (with $\varepsilon = 0.15$) shows a straight-line portion suggestive of critical behaviour. Although it is not unusual that event distributions show a deviation from a power-law outside a lower and an cut-off, it is often required that the distribution behaves linearly in the log-log plot for at least two decades which is not reached here. It is possible that we need to increase the particle numbers or the size of the region of interest to allow any power-law to become apparent over larger scale.

This figure gives limited evidence for a power-law of the form

$$y \sim x^{-2.3}. \tag{6}$$

If the straight line represents a true power-law relationship then we should suspect that it is possible to tune the system via a parameter variation from subcritical, through critical, to supercritical behaviours. Using the ε parameter we can tune the swarm behaviour in this manner, see the upper plots in figure 2. Large values ($\varepsilon = 0.5$) appear to result in a subcritical swarm with a shortage of large swarm movements, whilst a small value ($\varepsilon = 0.075$) results in an excess of large moves making the swarm appear supercritical.

As the intent is to remove the need to set any parameter values we set $\varepsilon = 1$ and σ equal to the size of the problem space. The PSO parameters are altered on each iteration and are given arbitrary initial values. Figure 3 shows an example of the log-log plot for this configuration to show that behaviour still maintains the mix of behaviours seen previously.

Stagnation The swarm dynamics suggest stagnation may be avoided, but we wish to see continued improvements are being found. In figure 4 we plot each fitness improvement found by the two approaches. For PSO, whilst there are many early improvements, these cease after a while. The number and ability to continue to locate further improvements is again a function of the problem and the chosen parameters. CriPS continues to locate improvements with continuing iterations.

Table 1 quantifies these results. Whilst fewer improvements are made by CriPS as it runs, it still continues to locate improvements, whilst PSO ceases to find better solutions.

Benchmark comparison Assessment of metaheuristic algorithms used to solve optimisation problems requires comparison across a number of problems. In recent years this is often achieved via competitions that present a range of benchmark problem functions with different characteristics. Such benchmarks may or may not capture the nature of real world problems, but seems a reasonable approach in the absence of alternatives.

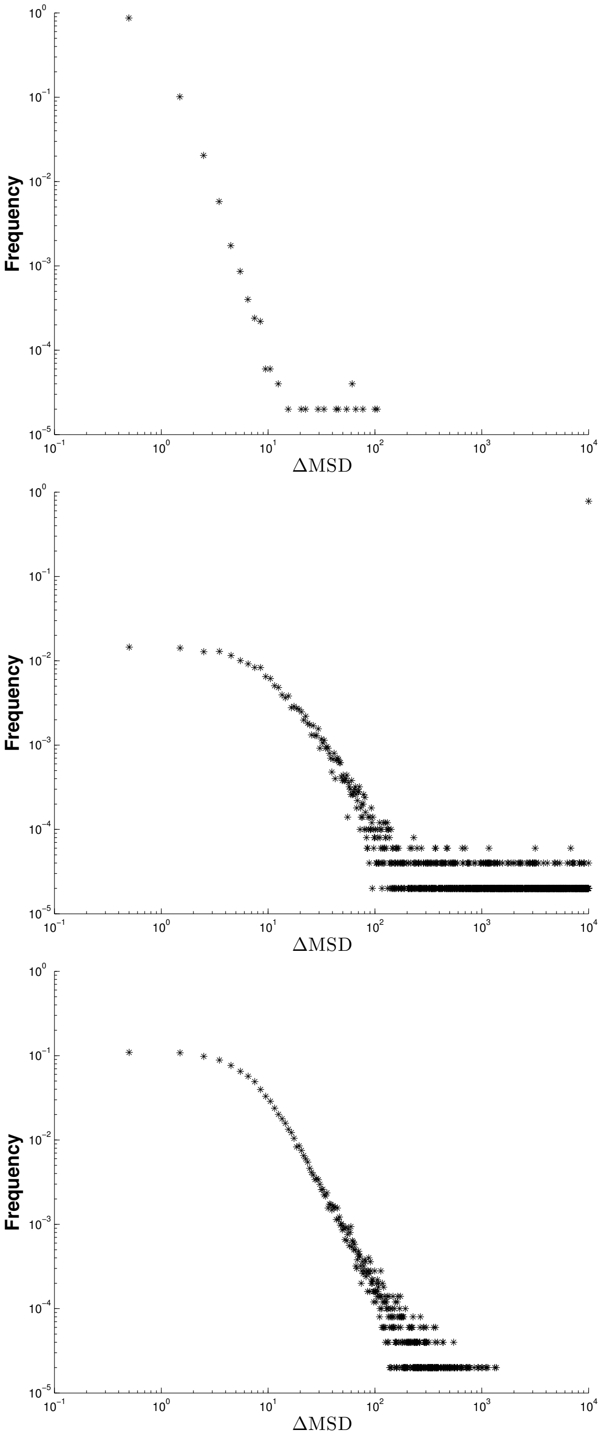


Figure 2: Different ε values result in differing mixtures of exploration and exploitation behaviours. Here we set σ to $\frac{1}{5}$ of the problem space's size. Top, supercritical, $\varepsilon = 0.5$. Middle, subcritical, $\varepsilon = 0.075$. Bottom, critical, $\varepsilon = 0.15$.

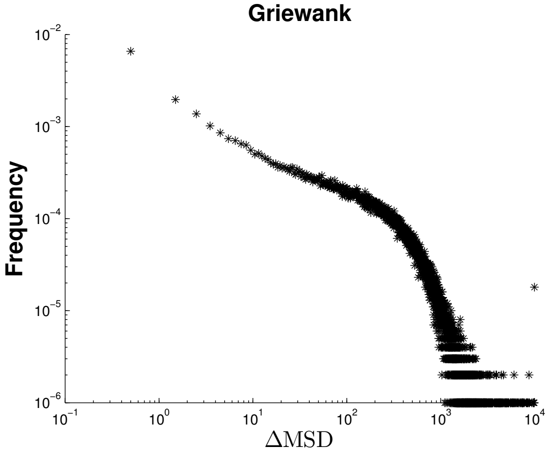


Figure 3: Distribution of swarm size changes for CriPS and Griewank problem space. Parameter σ is set to the problem space size, $\varepsilon = 1$, all PSO parameters set to arbitrary initial values.

Iteration range	**PSO**	**CriPS**
0-10000	2255.1 (342.1)	2499.6 (366.5)
10001-20000	4.1 (9.5)	1864.3 (921.4)
20001-30000	1.1 (0.55)	217.7 (449.7)
30001-40000	10.1 (45.8)	140.4 (540.2)
40001-50000	0 (0)	20.3 (82.4)

Table 1: Comparison of CriPS algorithm with our implementation of PSO. Each algorithm is run 30 times, for 50000 iterations. The mean and standard deviation (in parentheses) of the number of times an improved fitness is located was counted within each 10000 iteration period. In all cases the problems are defined as 30 dimensional, there are 30 particles in the swarm. Parameter ω was initialised to 0.815, α_1 and α_2 to 1. Additionally ε was set to 1 and σ set to the size of the problem space.

We adopt the protocol of the 2013 IEEE Congress of Evolutionary Computation (CEC2013)'s Real-Parameter Single Objective Optimization competition. Algorithms were run against 28 problem functions (with translations and rotations applied), 51 times in each of 10, 30 and 50 dimensions, with a maximum functional evaluation budget. A rank-sum approach evaluated the 21 competing algorithms. We have executed CriPS using the same protocol. For our comparisons the mean result we achieve against each function:dimension pair is compared to the matching values for the competition's algorithms that finished in positions 1, 5, 10, 15, 17, and 21. For each function:dimension pair we rank these results. The average rank across all pairs is used to rank the algorithms as shown in table 2. We estimate that we would have come roughly 16th in the 21 algorithms. This seems a reasonable result given that we are only using the standard PSO mechanisms to locate improvements.

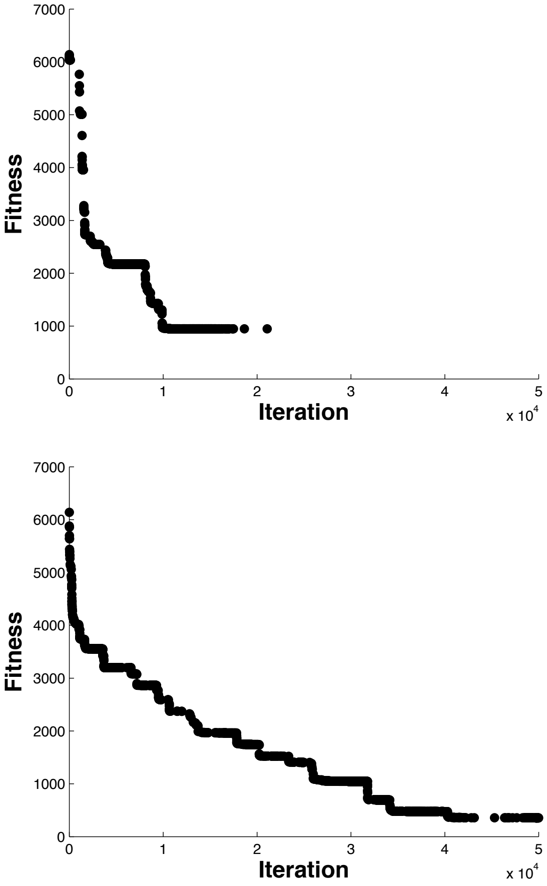


Figure 4: Fitness improvements found. Each point represents an improved solution found for the problem being explored (here the Schwefel function). Upper: Standard PSO, Lower: CriPS.

Algorithm	CEC 2013 position	Average ranking
NBIPOP-aCMA-ES	1	1.95
NIPOP-aCMA-ES	5	2.35
CDE:b6e6rl	10	2.88
JANDE	15	4.86
CriPS	n/a	4.96
TPC-GA	17	5.51
PLES	21	5.50

Table 2: Average ranking of CriPS algorithm in comparison with a selection of CEC2013 algorithms. Mean fitness obtained for each of the 84 function:dimension pairings are ranked. These rankings are averaged to obtain the result above. Comparator algorithm results and links may be found at `http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2013/CEC2013.htm`

In particular CriPS performed particularly poorly on the simpler problems. This is to be expected as even when the best course of action is simply to do gradient descent the dynamics of our algorithm will expend effort by causing expansions in the swarm. We note however that against one function (number 8: Rotated Ackleys Function) we ranked first. In a number of other problems we ranked in the top ten.

Discussion

Our CriPS swarm showed some limited evidence for criticality. Whilst the log-log plots appear to show straight-line sections, the range over which they manifest is too limited to draw concrete conclusions. However the algorithm does appear to achieve a balance of exploitation and exploration such that all problem space may potentially be visited. This dynamic allows the swarm to avoid stagnation. In addition no special parameter values need tuning. Values are either arbitrary or are set using a measure of the problem. In future it may be that feedback from the discovered manifold of the problem space (rather than the average velocity norm metric) may help to tune the algorithm to locate better solutions.

CriPS varies the PSO parameters synchronously. As we begin with non optimal parameter values there are large portions of the ω, α_1, α_2 parameter space that the algorithm can't use. We found that a CriPS variant that stochastically picks one of the three parameters to vary on each iteration appears to outperform our standard approach. We speculate that this is because the algorithm can exploit the full PSO parameter space.

We achieve reasonable results when compared with the CEC2013 competition. Performing in the top ten for a number of the objective functions specified (numbers 8, 9, 16, 19), notably winning on function 8 in all dimensionalities. However, our chief aim is to explore the possibility of engineering a critical dynamic in the PSO swarm to avoid the need to tune parameters and to avoid stagnation. CriPS uses standard PSO to guide it to locate better results. This is a limited factoring in achieving better results. Recent high performing metaheuristics e.g. CMA-ES, are more effective at guiding swarms to better solutions. These approaches still require decisions to be made regarding when the swarm should be restarted, and what population size to choose. We are combining a CriPS-like mechanism with covariance matrix update to allow CMA-ES to avoid the need to adopt a restart strategy.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council, grant number EP/K503034/1.

References

Bak, P. (1997). *How nature works*. Oxford University Press, Oxford.

Bak, P. and Sneppen, K. (1993). Punctuated equilibrium and criticality in a simple model of evolution. *Physical review letters*, 71(24):4083–4086.

Bak, P., Tang, C., Wiesenfeld, K., et al. (1988). Self-organized criticality. *Physical review A*, 38(1):364–374.

Bansal, J., Singh, P., Saraswat, M., Verma, A., Jadon, S. S., and Abraham, A. (2011). Inertia weight strategies in particle swarm optimization. In *Nature and Biologically Inspired Computing (NaBIC), 2011 Third World Congress on*, pages 633–640. IEEE.

Beggs, J. M. and Plenz, D. (2003). Neuronal avalanches in neocortical circuits. *The Journal of neuroscience*, 23(35):11167–11177.

Bratton, D. and Kennedy, J. (2007). Defining a standard for particle swarm optimization. In *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, pages 120–127. IEEE.

Chowdhury, S., Tong, W., Messac, A., and Zhang, J. (2013). A mixed-discrete particle swarm optimization algorithm with explicit diversity-preservation. *Structural and Multidisciplinary Optimization*, pages 1–22.

Clerc, M. (2010). *Particle swarm optimization*, volume 93. John Wiley & Sons.

Clerc, M. and Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *Evolutionary Computation, IEEE Transactions on*, 6(1):58–73.

Cordero, C. G. (2012). Parameter adaptation and criticality in particle swarm optimization. Master's thesis, The School of Informatics, The University of Edinburgh.

Eurich, C. W., Herrmann, J. M., and Ernst, U. A. (2002). Finite-size effects of avalanche dynamics. *Physical review E*, 66(6):066137.

Fernandes, C. M., Merelo, J. J., and Rosa, A. C. (2012). Controlling the parameters of the particle swarm optimization with a self-organized criticality model. In *Parallel Problem Solving from Nature-PPSN XII*, pages 153–163. Springer.

García-Villoria, A. and Pastor, R. (2009). Introducing dynamic diversity into a discrete particle swarm optimization. *Computers & Operations Research*, 36(3):951–966.

Jensen, H. J. (1998). *Self-organized criticality: emergent complex behavior in physical and biological systems*, volume 10. Cambridge university press.

Kennedy, J. (1998). The behavior of particles. In Porto, V., Saravanan, N., D.Waagen, and Eiben, A. E., editors, *Evolutionary programming VII*, pages 579–589. Springer.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948. IEEE.

Levina, A., Herrmann, J., and Geisel, T. (2007). Dynamical synapses causing self-organized criticality in neural networks. *Nature Physics*, 3(12):857–860.

Loshchilov, I. (2013). CMA-ES with restarts for solving CEC 2013 benchmark problems. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 369–376. Ieee.

Lovbjerg, M. and Krink, T. (2002). Extending particle swarm optimisers with self-organized criticality. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 2, pages 1588–1593. IEEE.

Olami, Z., Feder, H. J. S., and Christensen, K. (1992). Self-organized criticality in a continuous, nonconservative cellular automaton modeling earthquakes. *Phys. Rev. Lett.*, 68:1244–1247.

Qin, A. K. and Suganthan, P. N. (2005). Self-adaptive differential evolution algorithm for numerical optimization. In *IEEE Congress on Evolutionary Computation*, volume 2, pages 1785–1791. IEEE.

Richer, T. J. and Blackwell, T. M. (2006). The Lévy particle swarm. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 808–815. IEEE.

Riget, J. and Vesterstrøm, J. S. (2002). A diversity-guided particle swarm optimizer — the ARPSO. *EVALife Technical Report no. 2002-02*.

Shew, W. L., Yang, H., Yu, S., Roy, R., and Plenz, D. (2011). Information capacity and transmission are maximized in balanced cortical networks with neuronal avalanches. *The Journal of Neuroscience*, 31(1):55–63.

Shi, Y. and Eberhart, R. C. (1999). Empirical study of particle swarm optimization. In *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, volume 3. IEEE.

Wang, Y., Li, B., Weise, T., Wang, J., Yuan, B., and Tian, Q. (2011). Self-adaptive learning based particle swarm optimization. *Information Sciences*, 181(20):4515–4538.

Yang, X.-S. and Deb, S. (2009). Cuckoo search via Lévy flights. In *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pages 210–214. IEEE.

Zhan, Z.-H., Zhang, J., Li, Y., and Chung, H.-H. (2009). Adaptive particle swarm optimization. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(6):1362–1381.

Embodied Evolution of Artificial Cells in a Hybrid Wet/Hard-ware Platform

Juan Manuel Parrilla Gutierrez and Leroy Cronin

WestChem, School of Chemistry, University of Glasgow, Glasgow G12 8QQ, UK
Lee.cronin@glasgow.ac.uk

Abstract

The development of life over billions of years is the product of complex evolutionary processes between populations of living entities and the environment, together selecting fitter systems. Biological cells are outstanding examples of complex non-equilibrium systems showing emergent dynamics, and are the only example of naturally occurring, self-maintaining and self-replicating machines. How chemistry undertook the transition to biology is one of the most important questions relating to the origin of life and the design of artificial life beyond the biological paradigm. Theories focusing on RNA, proteins, metabolism, or protocells have been postulated so a key question is what basic infrastructure is required for simple chemical systems to show autonomous evolutionary dynamics.

Herein, we show the emergence of evolution in a hardware platform coupled with a wet ALife system (Gutierrez, 2014) (Figure 1). The hardware platform consists of a fully automated liquid handling robot capable of producing droplets in a Petri dish, equipped with a camera for video recording and image analysis. The platform was designed and built based upon the open source 'RepRap' 3D-printer (Jones, 2011), where the extruder was replaced by a liquid handling carriage with four different nozzles connected to syringe pumps for producing reagent mixtures inside a well plate, and an automated glass syringe to produce the droplets. RepRap's 3D printer controller software was also adapted for these new capabilities adding new functionality to control the liquid handling. The platform was connected to a computer where the algorithms and the image processing were processed.

The wet ALife system was designed as oil droplets generated in an aqueous phase environment (Hanczyc, 2007) (Browne, 2010). Our ALife system comprised of a four component system including: 1-Octanol, Diethyl-phthalate, 1-Pentanol and Octanoic acid, aiming for droplets that have motility, some stability, and a range of solubilities, densities, polarities and viscosities.

The artificial cell genotypes are composed by mixing four different chemical (oil) inputs embodied as populations of droplets within a specific environment, and a range of novel behaviors emerged which were characterized using a lattice search to describe the phenotypic space. Populations of these artificial cells were then evolved using an evolutionary algorithm informed by image recognition, exploring their ability to divide, move and vibrate. By fully automating the process it was able to produce large amounts of genetic algorithm data which enabled us to analyze and create a theoretical model which reinforces the hypothesis of oil droplets systems as artificial cell candidates.

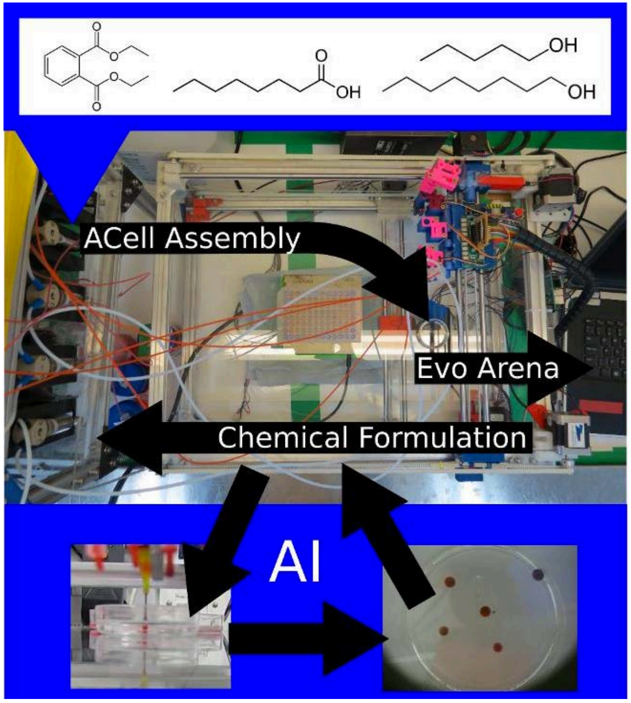


Figure 1: Hybrid hardware-wetware platform. Four different oil inputs acted as reagents for our artificial cells synthesis when place inside an aqueous phase. These reagents were handled by syringe pumps, mixed in a well plate, and generated inside the Petri dish using an automated glass syringe. Populations of four droplets were generated, and a camera recorded the experiment, which was later on analyzed in the basis of a defined fitness function. This information was then sent to an evolutionary algorithm, which decided the composition for a new experiment.

References

Gutierrez, J. M. P. *et al.* (2014). Evolution of oil droplets in a chemorobotic platform. *Nat. Commun.* 5:5571

Jones, R. et al. (2011). RepRap – the replicating rapid prototyper. *Robotica* **29**, 177-191.

Hanczyc, M. M. *et al.* (2007). Fatty acid chemistry at the oil-water interface. *J. Am. Chem. Soc.* **129**(30):9386-91.

Browne, K. P. *et al.* (2010). Self-Division of macroscopic droplets. *Angwe. Chem. Int. Ed*. **49**(38):6756-6759

Ontogeny and adaptivity in a model protocell

Eran Agmon[1,2], Alexander J. Gates[2,1] and Randall D. Beer[1,2]

[1] Cognitive Science Program, Indiana University, Bloomington, IN 47406, USA
[2] School of Informatics and Computing, Indiana University, Bloomington, IN 47406, USA
agmon.eran@gmail.com

Abstract

Viability, ontogeny, and adaptivity have been widely discussed within the context of emergent individuality. This paper provides an initial step towards a more formal treatment of these concepts. A network of possible ontogenies is uncovered by subjecting a model protocell to sequential perturbations, and mapping the resulting structural configurations. The analysis of this network reveals trends in how the protocell can move between configurations, how its morphology changes, and how the role of the environment varies throughout. Viability is defined as expected lifespan given an initial configuration. This leads to two notions of adaptivity: a local adaptivity that addresses how viability changes in plastic transitions, and a global adaptivity that looks at longer-term tendencies for increased viability. The mechanisms of a minimal adaptive transition are analyzed, and it is shown that these rely on distributed spatial processes rather than an explicit regulatory mechanism.

Introduction

Biological individuals are a special class of physical systems that counter the universal trend towards disintegration. In any given moment, a physical system can be described by its structural configuration — the spatial arrangement of components from which it is constituted. The states and locations of these components unfold dynamically, and whereas most configurations tend towards a uniform equilibrium, biological individuals are a unique subclass of physical system that persist as individuals. The theory of autopoiesis argues this viability comes from their closure of production; as a result of their intrinsic dynamics and material exchange with the environment, biological individuals produce and distribute the materials needed to stabilize themselves [10].

From the perspective of an individual, an environment appears as a probability distribution over possible perturbations. These perturbations, together with the individual's intrinsic dynamics, determine the individual's subsequent states. Thus, a perturbation can have one of three consequences: the individual can be unaffected (a robust transition), it can be changed to a different viable configuration (a plastic transition), or it can cross into the set of nonviable configurations and disintegrate (a destructive transition).

An individual that remains viable for any length of time experiences a sequence of perturbations that induce a corresponding sequence of configurational changes. In this paper, we refer to an unbroken trajectory through the set of viable configurations as an *ontogeny*; when the trajectory crosses the boundary of viability into a nonviable region, closure of production is broken and the ontogeny ends. Different sequences of perturbations have the potential to induce different ontogenies. If the set of possible perturbations is known, one can in principle map the entire network of possible transitions that an individual can undergo.

The *viability* of any configuration in an ontogenic network can be defined as the average number of perturbations it is from disintegration, weighted by the probability of those perturbations. If many perturbations are needed to destroy a configuration, then it is highly viable; if few are needed, then it has lower viability. *Adaptivity* can be defined as the change in viability that follows plastic transitions. Taken locally, *adaptive transitions* are plastic transitions in which viability is increased, and *maladaptive transitions* are those in which viability is reduced. A more global notion of adaptivity addresses whether there are trends towards increased viability across possible ontogenies. This could be achieved in a variety of ways, ranging from an explicit regulatory subsystem [3; 6] to more emergent processes.

In this paper, the relationship between viability, ontogeny, and adaptivity is investigated in the context of a model protocell that we recently proposed [1]. After reviewing the model and describing the environment in which we place it, the paper is organized as follows. First, we build upon a framework for exploring ontogenies as a network structure [4]. Applying this methodology reveals a rich complexity of ontogenic structure, which we characterize through a combination of graph-theoretic, morphology-based, and statistical measures. Second, viability is measured for all configurations across the ontogenic network, and increases in the measure allow us to identify trends of adaptive change. Finally, the mechanisms of an adaptive transition are analyzed in great detail.

A model of emergent individuality

An essential feature of biological individuals is their emergence from material components. To develop a theoretical understanding of such systems, the models must also display an emergence of individuality in the sense that they should exhibit metabolism-boundary co-construction.While such emergence has been explored in systems with abstract physics, such as the Game of Life [4], the model described here moves towards a more realistic chemistry.

The spatial model of molecular concentration dynamics considered here includes the diffusion, repulsion, chemical reactions, and decay of four molecular species: membrane (M), autocatalyst (A), food (F), and water (W). The chemical reactions are such that A is produced from an autocatalytic reaction between A and F, while M molecules are produced by a reaction which consumes both A and F. Both A and M also decay at a constant rate. Considered individually, molecular concentrations diffuse across a 2-dimensional lattice at a constant rate. However, the presence of repulsion between molecular types breaks the traditional symmetries. Based on the behavior of phospholipid bilayers, the model implements anisotropic repulsion between membrane molecules M and both A and W. This requires the introduction of a fifth state variable, θ, which defines the orientation of M and behaves with its own dynamic of alignment with neighboring orientations. Here, we utilize a 40×40 lattice. Each lattice point has the five state variables, resulting in an 8000-dimensional coupled dynamical system. For specific details of the model's implementation, refer to [2].

The model displays two distinct types of equilibrium points: the uniform equilibrium point with zero concentrations of both A and M, and stable inhomogeneities in which positive concentrations of A and M are maintained. The system is considered stable when the average temporal derivatives of A and M over the entire lattice satisfies $\frac{1}{2}\sum_{m\in\{A,M\}}\sum_{x_{ij}}|\dot{m}(x_{ij},t_k)| < \epsilon_1$ for all $t_k \in [t, t+1000]$ for a sufficiently small $\epsilon_1 = 0.05$. Even though these configurations are equilibrium points, they are still chemically active, with positive diffusion and reaction rates. It was demonstrated that, in the later class of equilibrium points, M was necessary to contain A at concentrations high enough such that A could continuously construct both A and M faster than their molecular decay. Thus, these configurations exhibit metabolism-boundary co-construction [2].

Throughout this work, we study the ontogeny of a particular configuration we named *SC* (stable configuration), shown on the left side of Figure 1. Due to $\theta's$ initialization, *SC* has broken symmetries across both its horizontal and vertical axes.

Environment as perturbation

From the individual's point of view, an environment is a probability distribution over a set of perturbations. We consider an environment $\mathcal{E}$, which consists of perturbations that increase the concentration of either autocatalyst (A) or membrane (M) in the local neighborhoods of nine distinct focal points. These locations are specified relative to the given configuration according to an algorithm that guarantees placement on the configuration (Figure 1). The increase in concentration is determined by a Gaussian function of the form: $G(x_{ij}) = \alpha e^{-(|x_{ij}-x_f|)^2/2\sigma^2}$, where $|x_{ij}-x_f|$ is the distance from the focal point x_f to the given cell x_{ij}, α is the magnitude of the function, and $\sigma^2 = 2.0$ determines its width. Four different magnitudes of $\alpha = [0.5, 1, 1.5, 2]$ are used. In total, environment $\mathcal{E}$ consists of 72 possible perturbations, each of which occurs with a uniform probability.

Once a perturbation is applied, it instantaneously displaces the system in state space. The system dynamics then unfold towards a limit set, resulting in one of three classes of possible outcomes: the uniform state, the same viable con-

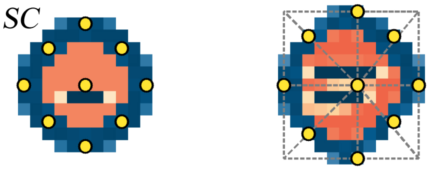


Figure 1: *SC* is the configuration on the left, which provides the starting point for this paper's exploration of ontogenic networks. These diagrams show only the autocatalyst (red) and membrane (blue) concentrations; food and water have been removed for clarity. The 9 relative perturbation locations from $\mathcal{E}$ are denoted by yellow dots. The right figure shows how locations are determined: a box encapsulates the configuration, and focal points are centered on the lattice cells according to intersections with the box's lines.

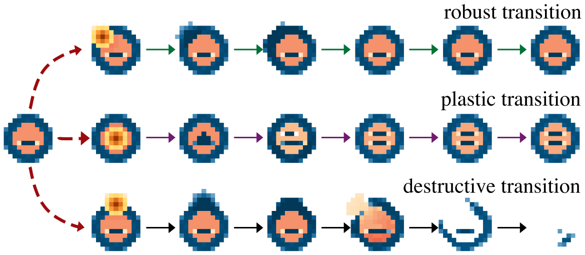


Figure 2: Three perturbations from environment $\mathcal{E}$ applied to *SC*. The perturbations increase the membrane concentration with the same amplitude $\alpha = 2$, as shown in yellow. Following perturbation, the system undergoes transients that stabilize in different attractor classes. The top branch shows a robust transition that returns to the original configuration, the middle branch shows a plastic transition that brings the system to a different stable configuration, and the bottom branch shows a destructive transition.

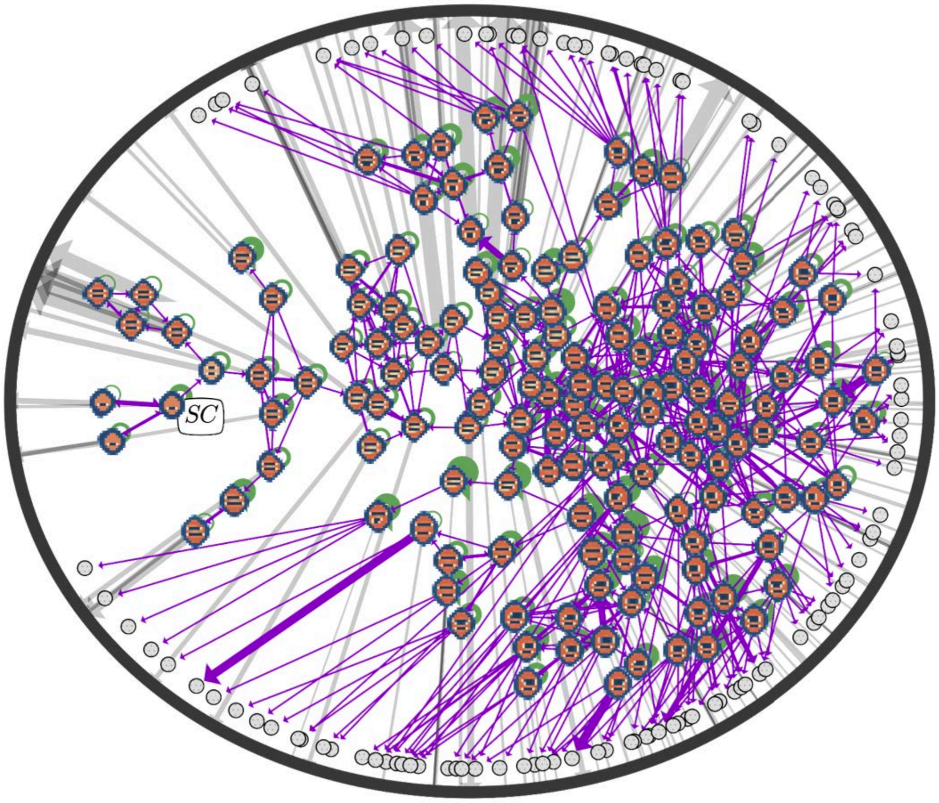


Figure 3: The ontogenic network *ON* captures the structure of all possible ontogenies starting at *SC* in environment $\mathcal{E}$. All configurations' morphologies are shown as nodes while the death state is depicted as a black ellipse. Unsearched configurations are denoted as gray nodes. Directed edges capture the transitions between configurations and are categorized as robust (green self-loops), plastic (purple edges), and destructive (gray edges). Edge width represents the log-probability for each transition.

figuration, or a different viable configuration. Examples of these three possibilities are illustrated in Figure 2.

To determine the identity of the resulting configuration, C, a comparison metric is used within the 3200-dimensional state space of M and A molecular concentrations (40x40 lattice sites for two molecule types). The structural distance from each previously observed configuration, C', is found by taking the summed absolute difference between the configurations' states $d(C, C') = \sum_{m \in \{A,M\}} \sum_{x_{ij}} |m_C(t, x_{ij}) - m_{C'}(t, x_{ij})|$. If this sum satisfies $d(C, C') < \epsilon_2$ with $\epsilon_2 = 1.0$, then we consider the configurations equivalent.

Ontogenic networks

Typical environments are the source of repeated perturbations which induce a sequence of changes to an individual's structure. A single trajectory through the set of viable configurations is an ontogeny. Different perturbations have the potential to induce different plastic transitions, resulting in different ontogenic trajectories. The structure of all possible ontogenies defines an *ontogenic network*, in which the viable configurations and death state constitute the network's nodes, and each environmental perturbation is a directed edge. Given a specific individual's configuration in a specific environment, the full ontogenic network is obtained by exhaustively characterizing the configuration's response to every environmental perturbation. The process is then repeated for all subsequent configurations until closure is achieved (i.e. every transition results in either a previously characterized configuration or death) [4].

The full ontogenic network formed by *SC's* repeated exposure to environment $\mathcal{E}$ is a multigraph with a set of reachable configurations as its nodes, each of which is the source for 72 directed edges. Following each perturbation, the system was given sufficient time to relax back to a stable condition before the next perturbation was applied. The network was generated by a breadth-first search that exposed all configurations to $\mathcal{E}$ as they were discovered. For this paper, the search was terminated at a uniform depth of 16 from *SC*. Configurations on the unsearched frontier are excluded from this section's analysis.

As a first step towards characterizing the structure of this network, we focus on the relationships between viable configurations. This suggests reducing the full ontogenic net-

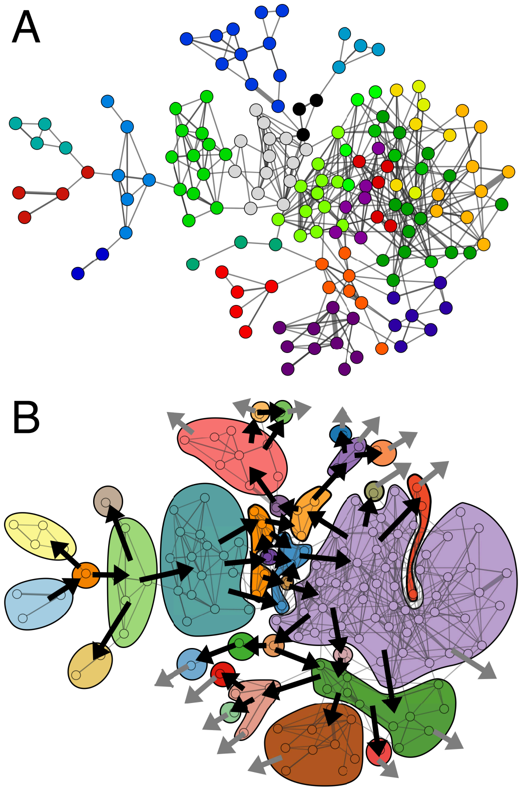


Figure 4: Two copies of *ON* (with the same layout as Figure 3). A) *ON*'s nodes are colored by cluster membership in InfoMap communities. B) Strongly connected components are shown as colored regions, with arrows indicating the direction of ontogenic change. Gray arrows indicate transitions to the unsearched frontier.

work by combining edges according to equivalence classes of transition outcome and considering the death state with all destructive transitions separately from the rest of the network. The resulting reduced ontogenic network (*ON*) is shown in Figure 3. *SC's* initial asymmetry is propagated through all configurations in *ON*. In theory, mirrored initial conditions can yield ontogenies identical to *ON*, but with mirrored configurations.

Remarkably, this pairing of a simple configuration and environment generates an extensive ontogenic network, rich with features. It has 154 stable configurations (represented in the figure by the morphological structure of A and M concentrations) as nodes while the death state is shown by the surrounding black ellipse. The directed edges in *ON* reflect the probability for a transition to occur given a random perturbation from the environment, with robust transitions shown as green self-loops, plastic transitions shown as directed purple edges between configurations, and destructive transitions shown in gray.

A statistical analysis of *ON* reveals that configurations vary widely in their response to perturbations from $\mathcal{E}$. Robust transitions occur with a probability from the range $[0.0, 0.7917]$ and a mean of 0.4715, destructive transitions occur with a probability from the range $[0.0972, 0.9583]$ and a mean of 0.3210, and plastic transitions, when they exist, occur with a probability from the range $[0.0139, 0.6528]$ and a mean of 0.0481. The probability for a configuration to plastically change is found as the sum over all of its plastic transitions. For the configurations in ON, the probability of a plastic change occurs in the range $[0.0, 0.8194]$ with a mean of 0.1682. It is interesting to note that all configurations have a non-zero probability of both destruction and survival in this environment. The out-degree distribution of the network reflects the number of different plastic options available to an ontogenic trajectory at each viable configuration. This distribution is supported in the range $[0, 9]$ with a mean of 3.4935. Further, 30% of plastic transitions are bi-directional; a number that indicates the network has many more bi-directional edges than found in a random graph. Indeed, this hypothesis is supported by a p-value $\ll 0.001$ when comparing *ON* to an ensemble of 1000 random graphs with the same number of nodes and degree-distribution.

The graph theoretic structure of the viable configurations in *ON* reflects several interesting characteristics of ontogenies starting at configuration *SC* in environment $\mathcal{E}$. First, the number of viable configurations increases exponentially with minimum path-length from *SC*. Second, *ON* has several distinct clusters as highlighted by InfoMap network community detection [12] (Figure 4A). These graph-theoretic clusters reflect sets of configurations for which transitions are more likely to remain within the set than leave it. Third, there are several configurations which function as bottlenecks for the network in the sense that ontogenies must pass through those configurations to reach different areas of the network. These configurations are determined by high values of betweenness centrality [11].

Strongly connected components (SCCs) found in the current *ON* demonstrate irreversibility, branching, and attractors (Figure 4B). SCCs are sets of configuration in which all configuration are mutually reachable [11]. There are 32 SCCs in *ON*. The presence of multiple SCCs indicates irreversibility — if the system moves from one SCC to another, it cannot return. Branching is illustrated by diverging paths from the SCCs. If an individual exits an SCC along one branch, the alternatives can no longer be explored. Finally, those SCCs with no outgoing connections are attractors in the sense that if an ontogeny enters one of these sets, it is guaranteed to remain there until death.

The analysis of ontogenies is further enriched by considering their morphologies — their unique spatial arrangements of molecular concentrations. Recall that each configuration defines a point in the 3200-dimensional state space of M and A molecular concentrations. Due to their high dimensionality, these configurations are projected onto a 2-dimensional manifold using the IsoMap manifold identifi-

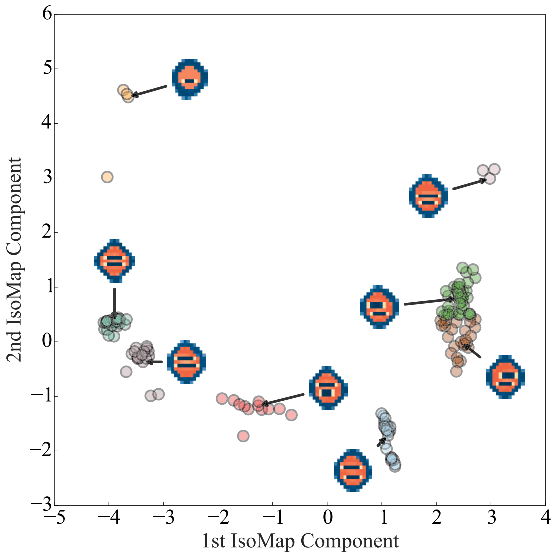


Figure 5: *ON's* configurations, clustered according to morphological similarity. Each point is a single configuration's morphological state projected onto a two-dimensional space determined by two IsoMap components. Different colors represent clusters detected by K-means clustering. An exemplar configuration from each cluster illustrates the cluster's unique morphological features.

cation technique [13] including the whole set of other configurations as neighbors. The resulting projection, shown in Figure 5, had a reconstruction error of 1.8374. This projection is indicative of morphological clustering with many points tightly grouped and relatively large distances between the groups. Specific partitions of configurations can be identified using K-means clustering [9]. The resulting 8 clusters are shown using different colors in Figure 5. Comparison of the exemplar configurations from each cluster, identified by their central position relative to the cluster's mean, demonstrates how the clusters vary along several qualitative dimensions: the size of the configuration, the shape of the outer membrane, the thickness of the outer membrane, and the number and arrangement of internal membrane structures.

Combining the morphological similarity of viable configurations with the transition network structure of *ON* reveals that plastic transitions are much more likely to occur between morphologically-similar configurations. A Bayesian BEST test [8] between the distribution of all inter-configuration distances using the Euclidean metric and the distribution of configuration distances for those linked by a plastic transition shows that the means of the distributions are distinctly different, with an average difference of 3.25 in a 95% confidence interval of $[3.16, 3.34]$. There is also a strong correspondence between the morphological clusters and the InfoMap clusters previously identified in *ON* as reflected by a normalized mutual information [5] value of 0.6746 between the two clusterings. Therefore, as an ontogeny unfolds and an individual falls into a graph-theoretic cluster, it also tends to maintain its morphological features by remaining within a corresponding morphological cluster.

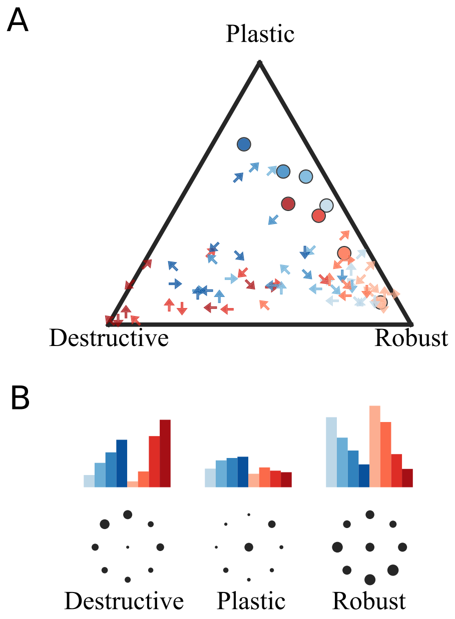


Figure 6: Environment $\mathcal{E}'s$ 72 perturbations positioned according to their probabilities of inducing a robust, destructive or plastic transition when applied to a random configuration from *ON*. A) Perturbations are shown in barycentric coordinates with markers illustrating their location, type, and size. Arrows indicate the perturbation location on the outer membrane and circles indicate perturbations to the central position. Color indicates molecular type, either membrane (blue) or autocatalyst (red). The perturbation's magnitude is shown by the darkness of the marker, with darker markers indicating larger magnitudes. B) Histograms categorize the probability of perturbations inducing a destructive, robust, or plastic transition conditioned on (upper) the perturbations type and magnitude or (lower) location.

In addition to characterizing attributes of the individual, the full ontogenetic network can also be used to characterize the influence of the environment. To proceed in the case of *SC* in $\mathcal{E}$, we return to the full ontogenetic network and classify each of the 72 perturbations by their probabilities of inducing a robust, destructive or plastic transition when applied to a random configuration from the set of viable configurations in *ON*. The resulting classification is visualized in Figure 6A according to barycentric coordinates for the three probabilities. Here, each point (arrow or circle) represents one of the 72 perturbations and the inverse distance between the point and each vertex reflects the associated probability of a transition in that equivalence class; a point directly on a vertex denotes 100% of transitions falling in that category.

Inspection of Figure 6A reveals that perturbations can vary widely in their consequences. Notably, all perturbations but one are destructive to at least one configuration, but none of the 72 perturbations are destructive to all configurations. There is a gradated tradeoff between perturbations' probabilities of inducing a robust transition versus inducing a destructive transition. A small subset of perturbations are associated with a large tendency to induce plastic transitions. These results can be further subdivided according to the location and magnitude of the applied perturbation as shown in Figure 6B. In the first row of the subfigure, three normalized histograms are shown which categorize perturbations based on outcome, molecular type (membrane in blue, autocatalyst in red), and magnitude (light to dark color). As one would expect, robust transitions are primarily associated with small perturbations while destructive transitions are associated with large perturbations. Plastic transitions occur more frequently for perturbations to membrane concentrations than to autocatalyst concentrations, regardless of magnitude. The second row of the subfigure illustrates three additional histograms categorizing perturbations based on outcome and location. These figures indicate that plastic transitions result more often from perturbations to the center and north-east locations while destructive transitions are slightly biased to perturbations on the north-west.

Quantifying viability and adaptivity

Viability is a consequence of well-matched configurations and environments. While some sequences of perturbations applied to an individual yield long ontogenies, the same individual exposed to different perturbations can result in shorter-lived ontogenies. The viability of a configuration is here defined as the expected lifespan over all of its possible ontogenies [2]. Two notions of adaptivity follow from this definition. Local adaptivity captures the change in viability resulting from a plastic transition: an adaptive transition increases a system's viability, a maladaptive transition reduces it. Global adaptivity is the general tendency for viability to increase with longer ontogenies.

In order to calculate viability, dynamics on an ontogenic network can be treated as a Markov chain. As a consequence of this, death becomes an absorbing state and most configurations are transients. Finding the average number of transitions from each transient state to an absorbing state is a well-studied problem [7], and gives our measure of viability. The lower bound on this measure is 1, which occurs when all perturbations bring the configuration to death within one step. The possibility of an immortal configuration would complicate this calculation by introducing infinite viability.

Applying these concepts to *ON* reveals a surprising abundance of adaptive transitions (Figure 7). The network's unsearched frontier is assumed to transition to death, giving a lower bound for the viability of all other configurations. A wide range of viabilities is found, from 1.0028 to 4.8382 with a mean of 2.1802, and 50% of the edges are locally adaptive. Interestingly, the most viable configuration in *ON* is not the most robust nor the one with the fewest destructive transitions. Its viability is a consequence of its embedding within the full ontogenic network.

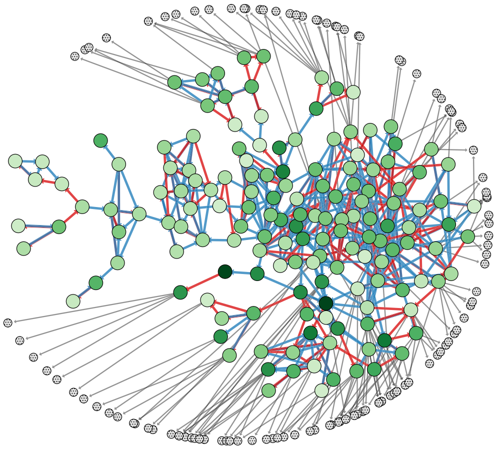


Figure 7: The network layout is the same as previous figures. Viability is shown by the darkness of the green nodes, locally adaptive transitions are blue, and locally maladaptive transitions are red.

Remarkably, ontogenies from *SC* also display global adaptivity. A correlation analysis found that a configuration's graph-theoretic path length from *SC* is positively correlated with its viability (r-value of 0.2882, p-value of 0.0003). This means that longer-lived ontogenies beginning at *SC* will generally experience an increase in expected lifespan. This global trend is configuration specific; each configuration in *ON* can have a different global adaptivity, and can even be globally maladaptive.

Mechanisms of adaptivity

This model provides an excellent opportunity to investigate the mechanisms underlying adaptivity. Previous formulations of adaptivity have assumed an explicit regulatory subsystem [3; 6]. This mechanism is designed to monitor the system's internal state relative to its boundary of viability and use this information to bring the system to more viable states. Models that implement this a priori assumption cover only a subset of possible mechanisms of adaptivity. In contrast, this paper's model demonstrates an emergent type of adaptivity. Analyzing the processes involved here can provide insights not previously possible.

As a first step towards understanding the mechanisms of emergent adaptivity, we examine a minimally adaptive scenario embedded within *ON*. An adaptive transition requires an environment of at least two perturbations, one which increases viability, the other which reduces it. Taken to its extreme, one perturbation would bring an initial configuration to death, and the other perturbation would bring it to a second, immortal configuration. Multiple instances of this exact scenario are found embedded within *ON* when environment $\mathcal{E}$ is restricted to only two perturbations.

The chosen example of a minimally adaptive scenario is shown in Figure 8A. The scenario begins with configuration α, which is subjected to the two perturbations. The perturbations are placed at different locations, but otherwise have the same magnitude and are both to the membrane field. On the top branch, α disintegrates, whereas on the bottom branch it undergoes a plastic transition to configuration β. β is then subjected to the same two perturbations, both of which result in robust transitions. In this minimal context, β can survive all perturbations, whereas α can survive only half of them. The transition from α to β is therefore adaptive.

One way to analyze this scenario is to utilize dynamical systems theory to characterize the system's phase space (Figure 8B). For visualization, the high-dimensional state space is projected onto the top two principal components. This makes trajectories appear to overlap, even though no overlap is possible in the full dynamics. Configurations α, β, and death are equilibrium points, each surrounded by a basin of attraction. The real basins of attraction are not visualizable, so faux basins of attraction are added to represent a hypothetical division of phase space. The two classes of perturbations (dashed red lines) instantaneously displace the system within the state space. Whereas perturbations to α move the system either to death's basin of attraction or to $\beta's$ basin of attraction, when applied to β they displace the system within the same basin of attraction. Given this analysis, adaptivity is explained by the congruence between the two configurations' basins of attractions and the available perturbations. The position of the equilibrium points, and the shape of their basins are such that the perturbed states fall outside of $\alpha's$ basin, yet remain within β's.

A more detailed investigation of mechanism needs to address the specific physico-chemical interactions that take place during this adaptive transition. A spatial analysis is here approached by taking sequential snapshots of transient configurations throughout the minimally adaptive scenario and identifying the critical differences in their morphologies (Figure 8C). First, we look at the adaptive transition from α to β (Figure 8C(1)) and the divergence from α that occurs throughout this transition (Figure 8C(2)). The sequence begins with α, at which there is no difference. Next, the perturbation displaces the membrane field, seen as a circular difference in the top right. Further down the sequence, the membrane concentration spreads around the boundary. While the north-east side of the configuration remains at increased concentrations, the rest of the boundary is slightly lower from its initial concentrations; this trait stabilizes at β. The most obvious difference between stable configuration β and α is a local increase in membrane concentration, which makes β rounder and thickens its boundary (arrow I).

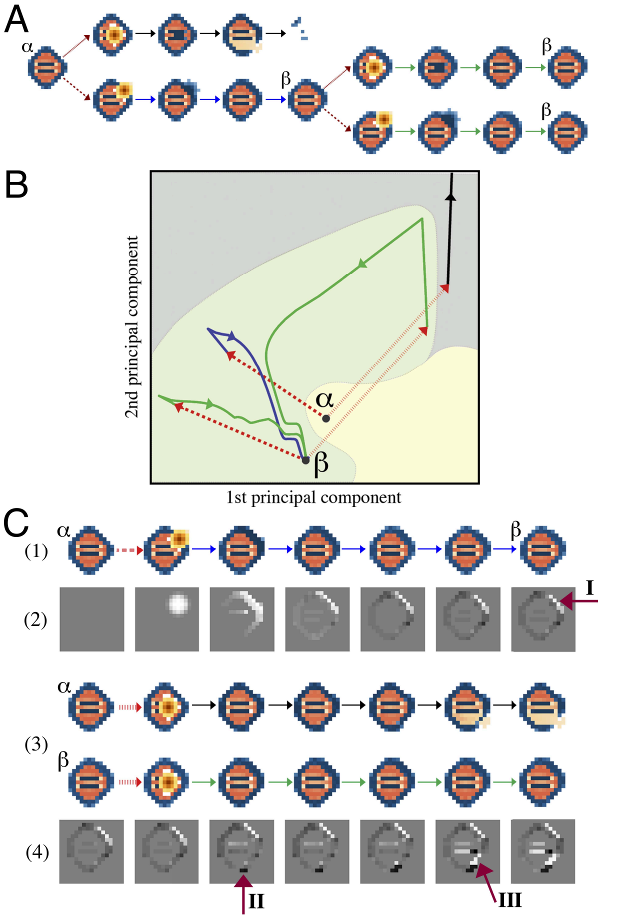


Figure 8: A minimally adaptive scenario and its analysis. A) Configuration α branches off into two transients following the application of two perturbations, one of which leads to disintegration, and the other to a plastic transition resulting in β. β is then perturbed by the same two perturbations and recovers, proving it has adapted. B) The same sequence of events shown in state space projected onto the top two principal components. Faux basins of attraction (yellow, green, gray regions) are added for explanatory purposes. Perturbations are indicated by the two classes of dashed red lines. Transients are shown as trajectories, with the destructive transition as a black trajectory, the adaptive plastic transition as a blue trajectory, and the robust transitions as green trajectories. C) The scenario is analyzed as sequences of spatiotemporal configurations. (1) The transition from α to β. (2) The difference from $\alpha's$ membrane field for the corresponding sequence in (1); the figure is gray where membrane concentrations are the same as α, white where they are more than α, and black where they are less than α. (3) Both α and $\beta's$ responses to the second perturbation. (4) The difference between membrane concentrations for the two sequences in (3); where membrane concentrations are the same in these sequences, the figure is gray, where the transient following β has a higher membrane concentration, the figure is white, and where it has lower concentration, the figure is black.

Next, we examine how these morphological differences allow β to survive a perturbation that α does not survive (Figure 8C(3)). The difference between these two sequences shows where the relevant divergences begin (Figure 8C(4)). The sequence begins with the initial difference between α and β. When the perturbations are applied to both configurations, they do not appear in the difference plot because the perturbations are spatially aligned. Later, the behavior begins to diverge. The obvious difference between their initial structures (arrow I) is not where the fatal divergence begins. Instead, a region on the bottom of α (arrow II), which bulges out and to the right, is the important feature. This growth draws $\alpha's$ membrane downward, reducing membrane concentration in the south-east boundary (arrow III). This opens up a tear in $\alpha's$ membrane, from which autocatalyst pours out and ultimately brings about disintegration, whereas β's morphology allows it to stay intact and restabilize.

In contrast with previous formulations of adaptivity, we see that no explicit regulatory mechanism is found during this emergent adaptive transition. Adaptivity is the result of distributed processes, and is better explained by their emergent spatio-temporal dynamics. The analysis demonstrates that local changes in chemical distributions, such as thickening membranes, have consequences for subsequent behavior. In dynamical systems terms, adaptivity is determined by the shape of configurations' basins of attraction, and how interactions with the environment move a system through the phase space. A transition is adaptive if it brings the individual to a configuration with a more accommodating basin.

Discussion

This paper marks the first analysis of ontogeny in a spatial chemical model that supports emergent individuality. The ontogenic network is a unique consequence of the individual's morphology paired with a particular environment. A combination of statistical and graph-theoretic methods revealed a rich structure, which includes clusters and bottlenecks that constrain ontogenic change. The reachable morphologies were found to cluster according to morphological similarity, and we showed that as an individual falls into a graph-theoretic cluster it tends to maintain its morphological features. Two notions of adaptivity followed from the definition of viability as average expected lifespan; local adaptivity looks at the change in viability resulting from plastic transitions, and global adaptivity looks at longer-term increases in viability. There was an abundance of local adaptivity within the ontogenies, and surprisingly, the model also displayed global adaptivity. Finally, the mechanisms of a minimally adaptive scenario were analyzed, demonstrating how adaptivity can be explained by distributed process rather than explicit regulatory mechanisms.

The combination of this model and analytical techniques provides a foundation for studying the emergence of viability, ontogeny, and adaptivity in more biologically realistic systems. One natural extension along these lines reconceptualizes viable configurations as members of dynamically richer limit sets. Another recognizes that environments also include sequential perturbations that occur on similar or faster timescales compared to an individual's intrinsic dynamics; these environments would keep the system from stabilizing at a limit set and require a new, continuous conceptualization of ontogenic change. Further, biological individuals and environments are structurally coupled [10], suggesting that an individual's behavior could induce correlations in its environment which affect future interactions. All of these factors need to be considered in a generalization of adaptivity to real-world biological individuals.

Acknowledgements

This work was supported by NSF grant IIC-1216739, and Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute, and by the Indiana METACyt Initiative. The software is freely available at https://github.com/eagmon/co-construction_base

References

[1] Agmon, E., Gates, A. J., Churavy, V., and Beer, R. D. (2014). Quantifying robustness in a spatial model of metabolism-boundary co-construction. *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, 14:3–5.

[2] Agmon, E., Gates, A. J., Churavy, V., and Beer, R. D. (in press). Exploring the space of viable configurations in a model of metabolism-boundary co-construction. *Artificial Life.*

[3] Ashby, W. R. (1960). *Design for a Brain*. Springer.

[4] Beer, R. D. (2014). The cognitive domain of a glider in the game of life. *Artificial life*, 20(2):183–206.

[5] Danon, L., Daz-Guilera, A., Duch, J., and Arenas, A. (2005). Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(09):P09008.

[6] Di Paolo, E. A. (2005). Autopoiesis, adaptivity, teleology, agency. *Phenomenol Cogn Sci*, 4(4):429–452.

[7] Grinstead, C. M. and Snell, J. L. (1998). *Introduction to Probability*. American Mathematical Soc.

[8] Kruschke, J. K. (2013). Bayesian estimation supersedes the t test. *Journal of Experimental Psychology: General*, 142(2):573.

[9] Lloyd, S. (1982). Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137.

[10] Maturana, H. R. and Varela, F. J. (1980). *Autopoiesis and cognition: The realization of the living*. Number 42. Springer.

[11] Newman, M. E. (2003). The structure and function of complex networks. *SIAM review*, 45(2):167–256.

[12] Rosvall, M. and Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *PNAS*, 105(4):1118–1123.

[13] Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323.

Chemical Architectures for Self-Replicating Proto-Cells

Erwan Bigan[1,2], Jean-Marc Steyaert[1] and Stéphane Douady[2]

[1]École Polytechnique, 91128 Palaiseau Cedex, France
[2]Université Paris Diderot, 75205 Paris Cedex 13, France
bigan.erwan@orange.fr

Abstract

Two questions arise when considering chemical architectures for cellular life: (i) are specific features such as cycles, autocatalysis or metabolic closure required? (ii) how can chemical insulation between the inside and the outside be achieved?

As most theoretical biology work has focused on the inside of the cell, significant work has only been devoted to the first question, yet without any definite answer. On the one hand, specific chemical features such as cycles, autocatalysis or metabolic closure were either made explicit in previous theoretical models, or emerged as the result of complexity. On the other hand, artificial life efforts rely upon much simpler chemistries, typically consisting of only a couple of constituents and reactions.

We show that self-replication of a chemical system is possible without any explicit feature, and without the need for their emergence through complexity. We use a simple generic proto-cell model, whereby conservative chemical reactions occur within a membrane resulting from the self-assembly of one of the chemical species (membrane precursor), with this membrane being semi-permeable by diffusion to some other species (nutrient). We have mathematically proven simple minimal conditions pertaining only to the topology of the embedded chemical reaction network (CRN), and relying upon the concepts of moieties and siphons. Moities are conserved quantities and siphons are subsets of species whose absence cannot be compensated by the chemical reactions. A necessary condition for stationary growth is that each moiety must be fed. And a sufficient condition is that each siphon be fed (Bigan et al., 2014). Generating random CRNs with arbitrary stoichiometry, we find that these conditions are typically reached at a relatively low complexity, with a number of reactions only slightly above the number of species (Bigan et al., 2015b). When the necessary condition is met but the sufficient one is not, the proto-cell may reach a stationary growth regime or not depending on the kinetics of *pass* reactions, connecting a siphon to its complement and transferring mass into the siphon.

Siphons and *pass* reactions are useful concepts to address the second question about chemical insulation between the inside and the outside of the cell. The chemical composition of modern evolved cells is quite different from that of the outside growth medium, and often exhibit higher osmotic pressure inside than outside (e.g. bacteria or yeast with a cell wall surrounding the plasmic membrane). Previous experimental work has also shown that a higher inside osmotic pressure increases the proto-cell growth rate, thus giving a selective advantage.

Assuming that the same chemistry operates on both sides, we propose two chemical schemes that are capable of sustaining different compositions across a semi-permeable membrane with a higher inside osmolarity (Bigan et al., 2015a). The first scheme relies upon the concept of moieties and associated degrees of freedom, by having only some degrees of freedom fixed by the permeating species. A shortcoming of this first scheme is that all species are present on both sides of the membrane (albeit at different concentrations) including the membrane precursor. As a result, the membrane surface area (and thus the cell volume) grows by incorporation of membrane precursor from both sides of the membrane. This significantly reduces the capability for higher osmolarity. The second scheme relies upon the concept of siphons, by having all species present inside but only those outside the siphon present outside. Growth medium contamination by any lysed cell is alleviated if the *pass* reaction (transferring mass into the siphon) is catalyzed by the membrane surface. The ultimate insulation between the inside and the outside of the cell is achieved when this *pass* reaction only occurs across the membrane, which is active transport. This suggests an evolutionary path for the emergence of active transport.

From this, we make the following architectural recommendations for a stationarily growing proto-cell: CRN exhibiting a shorter siphon (shorter than the full set of species); membrane precursor in the siphon, and self-assembling in a structured membrane; structured membrane semi-permeable by diffusion to some nutrient outside the siphon, and connecting to the siphon via a *pass* reaction catalyzed by the membrane.

References

Bigan, E., Steyaert, J.-M., and Douady, S. (2014). On necessary and sufficient conditions for proto-cell stationary growth. In *Proceedings of the fifth International Workshop on Static Analysis and Systems Biology (SASB 2014). In press at* Electronic Notes in Theoretical Computer Science. Available at `http://arxiv.org/abs/1411.6772`.

Bigan, E., Steyaert, J.-M., and Douady, S. (2015a). Chemical schemes for maintaining different compositions across a semi-permeable membrane with application to proto-cells. Accepted for publication in *Origins of Life and Evolution of Biospheres*.

Bigan, E., Steyaert, J.-M., and Douady, S. (2015b). Minimal conditions for proto-cell stationary growth. *Artificial Life*, 21.

A population of information processing objects as a model of evolutionary dynamics

Richard J Carter*[1,2], Stephen Mann[1,2], Karoline Wiesner[2,3]

[1]Centre for Protolife Research, School of Chemistry, University of Bristol, UK.
[2]Bristol Centre for Complexity Sciences, University of Bristol, UK.
[3]School of Mathematics, University of Bristol, UK.
* Corresponding author: rich.carter@bristol.ac.uk

An information-centric approach to modelling the evolution and complexification of a population of interacting pre-biotic entities is presented. Based on previous work by Crutchfield and Görnerup (2006) we develop an agent-based model of information processing objects represented as finite state transducers that interact and replicate. Those agents that can process information cost effectively and that are well connected within an interaction network are deemed to have a higher degree of fitness. The potential trade-off between cost and connectedness of an agent is measured quantitatively by determining its structural complexity and the interaction network complexity of the population.

Simulations of the model examine whether there is a critical interplay between the optimisation of individual agents at the expense of a population-wide benefit versus the optimisation of cooperation and coupling of objects for mutual benefit at the expense of the individual agents (a form of autopoietic system as defined in Maturana and Varela (1980)).

An increased theoretical understanding of the processes that lead to dynamically stable structures and their evolutionary transitions in populations of informational processing objects could provide important insights in a range of disciplines including the emergence of viable protocell communities on the early Earth, design and synthesis of protocell populations in the laboratory, evolution of multicellularity, and genesis of cancerous tumours.

References

Crutchfield, J. P. and Görnerup, O. (2006). Objects that make objects: the population dynamics of structural complexity. *Journal of The Royal Society Interface*, 3(7):345–349.

Maturana, H. and Varela, F. (1980). *Autopoiesis and cognition: the realization of the living*. Springer Science & Business Media.

Evolving strategies for single-celled organisms in multi-nutrient environments.

Dominique Chu[1] and David J. Barnes[1]

[1]School of Computing, University of Kent, CT2 7NF, Canterbury, UK
dfc@kent.ac.uk

Abstract

When micro-organisms are in environments with multiple nutrients, they often preferentially utilise one first. A second is only utilised once the first is exhausted. Such a two-phase growth pattern is known as *diauxic growth*. Experimentally, this manifests itself through two distinct exponential growth phases separated by a lag phase of arrested growth. The duration of the lag phase can be quite substantial. From an evolutionary point of view the existence of a lag phase is somewhat puzzling because it implies a substantial loss of growth opportunity. Mutants with shorter lag phases would be prone to outcompete those with longer phases. Yet in nature, diauxic growth with lag phases appears to be a robust phenomenon. We introduce a model of the evolution of diauxic growth that captures the basic interactions regulating it in bacteria. We observe its evolution without a lag phase. We conclude that the lag phase is an adaptation that is only beneficial when fitness is averaged over a large number of environments.

Introduction

When bacteria (and other organisms) are presented with two (or more) nutrients of different kinds then they often show what is called, diauxic growth (Deutscher, 2008; Brückner and Titgemeyer, 2002). The discovery of this effect goes back to (Monod, 1949) and means that a cell takes up one nutrient first and exclusively. Only when this first nutrient has run out will it turn to the second. This first is usually preferred and affords higher growth. In wet-lab experiments diauxic growth is a well known, frequently encountered effect. In growth experiments it manifests itself through a signature fast initial exponential growth, followed by an episode of reduced or no growth, and then a second phase of exponential growth. The first growth phase is fuelled by the preferred nutrient; the second growth phase by the less preferred one. The phase of arrested growth in between is usually called the *lag phase* and can be very long compared to typical bacterial generation times. For example, in the case of the *E.coli* and glucose-lactose growth, the lag phase is of the order of a typical generation time (about 20 minutes).

Within biosciences, diauxic growth is a well known phenomenon. Its mechanisms are well understood and certainly its experimental phenomenology is routine for the working micro-biologist. Yet here we are interested in exploring the evolutionary origin of the effect. The standard account for diauxic growth is that it enables cells "to increase their fitness by optimizing growth rates in natural environments providing complex mixtures of nutrients" (Stülke and Hillen, 1999). This intuition is likely right. Ultimately, all it states is that diauxic growth helps to maximise growth, which, at least in the context of bacteria, equates to saying that it is a beneficial adaptation. However, what it does not state is how and in what way diauxic growth helps to maximise growth. What precisely are the conditions under which a two-phase growth strategy is better than simultaneous uptake of both nutrients?

One of the aspects that is particularly puzzling from an evolutionary point of view is the presence of the lag phase. Arrested growth during an exponential phase is costly, especially if it takes as long as a typical doubling time. It is not clear how this long phase of delay can be reconciled with the justification of diauxic growth as a strategy that maximises growth. Naively, one would assume that there are a number of strategies that could avoid the lag phase and still reap the benefits of diauxic growth. For example, a mixed strategy that relies on a small amount of uptake from the second nutrient while the first is still available. Or, a more sophisticated regulatory mechanism where uptake of the second nutrient is activated as the first nutrient runs out. This suggests that it is worthwhile to re-consider the evolutionary origin of diauxic growth.

Recently, there has also been some renewed experimental interest in diauxic growth. New methods in single-cell observation now make it possible to see what happens in diauxic growth at the level of the individual cell. There is now evidence (Boulineau et al., 2013) that the growth phases and the lag phase at the level of the individual cell are not as well defined as they are at the population level. For example, throughout the (population level) lag phase, some of the individual cells continue to grow without delay while others cease growth for substantial amounts of time before resuming growth well into the second growth phase. This suggests that diauxic growth is a population level effect, while at the level of the individual the dynamics are determined by noise and heterogeneity of the population. In turn, this suggests that in order to understand diauxic growth and its evolution it is necessary to combine population level dynamics, stochastic modelling and artificial evolution. This will then make it possible to understand which strategies emerge under which assumption.

To this end, we created a program that combines biochemical simulation techniques with agent-based methods. It can simulate populations evolving cell populations. The model we created with the program simulates each individual cell using a discrete stochastic simulation algorithm. Cells in the simulation are exposed to external nutrient that they need to take up and metabolise to grow. Once they have grown to a sufficient size they will divide. Over time a population of agents emerges. We also experimented with two types of evolution: an explicit approach using a genetic algorithm, and an implicit fitness approach. In the implicit approach, we monitor multiple concurrently-evolving populations with limited interaction/exchange between them.

In our model we simulate a scenario where two nutrients are offered to the cells: `N1`, which is the high quality nutrient, and `N2` which gives only half of the energy per molecule when compared to `N1`. For all our simulations we kept the structure of the internal cell dynamics fixed and only allowed the system to change regulation by adjusting parameter values. We find a hybrid strategy evolving: While there is a clear diauxic shift observable, we do not see an initial phase of growth fuelled solely by one nutrient. Instead, during the initial phase there is a certain amount of uptake and usage of both nutrients. However, the better quality nutrient, is taken up at a higher rate. Once the better nutrient is exhausted, the uptake rate of the second is increased strongly. Crucially, this happens without any delay—there is no lag phase evolving in our simulations and it depends crucially on the uptake/metabolism being capacity limited. This is in line with the intuition mentioned above: that a lag phase comes at a substantial fitness cost.

These results suggest that diauxic growth is a general phenomenon that will evolve as long as there is a limitation on the capacity of the uptake system. However, our results contrast with the strictly sequential nature of diauxic uptake in real bacteria. We hypothesise that this needs to be explained by life-time averaging of fitness.

Methods and Models

The Model of a cell

We used a minimal model of a cell that can take up two types of external nutrient: `N1, N2`. The internalised nutrients are `E1, E2`. They can be metabolised into internal energy `E0` with rates `lk1` and `lk2` respectively. This energy is required in order to express the porins `P1` and `P2`. These are required to take up `N1` and `N2` respectively. Furthermore, internal energy `E0` is required in order to produce biomass and growth. Growth is essential because the cell is only allowed to divide if it has reached a minimum size. As such, this model captures the fundamental trade-off that real cells also face: How much energy should be invested into growth and how much into maintaining the machinery necessary to take up nutrient? If everything is invested into growth then the cell will soon end up in a position where it cannot replace its uptake machinery. On the other hand, if it invests all into the uptake machinery, but none into growth, then it will not be able to divide and hence will be evolutionarily unfit.

Here we are interested in the dynamics of diauxic growth. Hence, we built into our model an interaction mechanism between uptake of nutrient 1 and nutrient 2. We modelled this mechanism loosely on the PTS uptake system in bacteria (Boianelli et al., 2012). Nutrients 1 and 2 can only be taken up by porins specific to them. Uptake of the preferred nutrient leads to the dephosphorilisation of a repressor R^*. Its dephosphorylised form `R` can bind with porin `P2`, thus blocking uptake of `N2`. The porin-repressor dimer disintegrates with a rate of `ukb`, thus restoring the ability of the porin to take up nutrient again. Note that R^* is not explicitly modelled but assumed to exist in constant abundance.

Cells divide with a given rate of 1 once they have reached a certain size threshold. Upon division the protein and porin content of the cell is randomly (but typically not equally) divided between parent and offspring. In our model we have no explicit degradation of proteins or porins. Instead we rely on dilution through division as a mechanism to control particle numbers per cells. This choice is, on the one hand, realistic with respect to real bacteria but it also reduces the model complexity by saving two additional parameters.

Restrictions on uptake mechanisms In this contribution we considered two variants of the model. One where the cell is limited in its capacity to take up/metabolise nutrients. Here, we represent this by a limit on the number of porins that the cell can incorporate into its surface. This is represented as a negative feedback from the number of porins to rate of porin expression. In the formal description of our model in Table 1 this restriction is implemented by the second term of the rate in the expression of `P1` and `P2` (sixth and seventh reactions). Below, we will also report simulations where the restriction on the porin number is disabled. This is realised simply by removing the second term.

Evolution

We considered two types of evolution: implicit and explicit. For both we varied the following parameters from Table 1: `lk1,lk2,K1,K2,leak1,leak2,g,KG,dR,kb,ukb`. All parameters are allowed to vary in the arbitrarily chosen but constrained floating-point interval $[0, 15]$. Only the parameters `dR, kb, ukb` regulating internal signalling reactions are multiplied by 10 before being used. This reflects that they are are happening on a faster time-scale

Description	Substrate	Product	Rate
`N1` taken up	`N1`	`E1 + R`	`[P1 hill(N1/volume,K,2)]`
`N2` taken up	`N2`	`E2`	`[P2 hill(N2/volume,K,2)]`
`E1` converted to energy	`E1`	`E0`	`[lk1]`
`E2` converted to energy	`E2`	`E0`	`[lk2]`
`E2` loss through inefficiency	`E2`	$\emptyset$	`[lk2]`
Porin 1 production	`E0`	`P1`	`[(leak1 + hill(E1,K1,2)) . (1-hill(P1+P2,surfaceArea/50,4))]`
Porin 2 production	`E0`	`P2`	`[(leak2 + hill(E2,K2,2)) . (0-hill(P1+P2,surfaceArea/50,4))]`
De-phosphoryilisation of repressor	`R`	$\emptyset$	`10 dR`
Repressor blocking porin 2	`R + P2`	`B`	`10 kb`
Decay of repressor-porin compound	`B`	`R + P2`	`10 ukb`
Volume increases by one unit (growth)	`5 E0`	`volume`	`g`
Division happens when volume reaches 50	`50 volume`	Division	`1`

Table 1: The definition of the model. The expression `hill(x,y,z)` abbreviates the function $x^z/(x^z + y^z)$. The last column gives the rate (when the expresssion is in square brackets) and the rate constant otherwise.

than gene-expression. The implicit scenario consisted of 32 environments. Each is populated by 5 random cells. A certain amount of N_1 and N_2 was given to each environment. Within each environment the cells then grew, took up nutrients, and divided when they reached the division threshold volume (set to 50). On division, cell volume and contents were randomly divided between the parent and offspring. The offspring cell underwent a mutation with a probability $P_m = 0.005$. Mutations are adjustments of the parameters by adding/subtracting up to 10 percent from the current value of the parameter (while respecting the constrained ranges.) Hence, over time the diversity of the population grows within each environment. Once all nutrient is exhausted the simulation of the environment is halted.

We simulated the environments in parallel and once all had exhausted their nutrient we performed a migration step. In all environments we first chose a small number of migrant cells, removed those from the population and then pruned the remaining population in each environment back to 5 cells by randomly discarding cells. Following this we chose random destinations for the migrant cells. Migrant cells replaced a random member of the existing seed population. The migration probability was set to 0.001 per iteration per cell. On average, this resulted in less than one migrant per environment with the settings we used. Once these steps were completed, each environment was simulated again and the process repeated for the desired number of generations.

Explicit evolution was implemented by using a standard GA with fitness proportional selection and a population size of 32 environments. The GA was run for 300 generations. Each solution in the population was represented by the parameter values of the model. Each environment was seeded by a single cell that was then grown on a mix of nutrients as explained above. Once a generation was completed, the final population size was recorded and used as the fitness for the corresponding solution.

In this article we report a competitive GA. This means that after a first GA evaluation of 300 generations, we restarted the evolutionary process. Yet, this time, we seeded each environment with a candidate solution (which evolves) *and* the best solution found during a previous evolutionary run. We only evaluate the fitness for the candidate solution. The fixed solution only serves as a competitor against which the evolving solutions have to persist. We performed several series of these competitive GAs whereby, in each iteration, the candidate solutions are evolved against the best solution emerging from the immediately preceding competition.

The Modelling Software

The simulation was performed using custom software developed in Java to support the general modelling and evolution of cellular species whose dynamics are defined by a parameterised set of reaction equations. This software combines agent-based modelling with stochastic simulations of biochemical networks. The user only needs to define the reaction network of one particular cell/organism and the number of instantiations of this cell. The software then automatically creates the relevant reaction networks to simulate each of the distinct cells.

A model is realised by an efficient implementation of the Gibson/Bruck stochastic simulation algorithm (SSA) (Gibson and Bruck, 1998) — a variant of Gillespie's algorithm (Gillespie, 1977). Since each cell in an environment potentially represents a separate parameterisation of the model's equations, and the population size will vary within a single generation, the number of reactions to be managed is dynamic and can grow very large. A pure implementation of Gillespie's SSA would recalculate the propen-

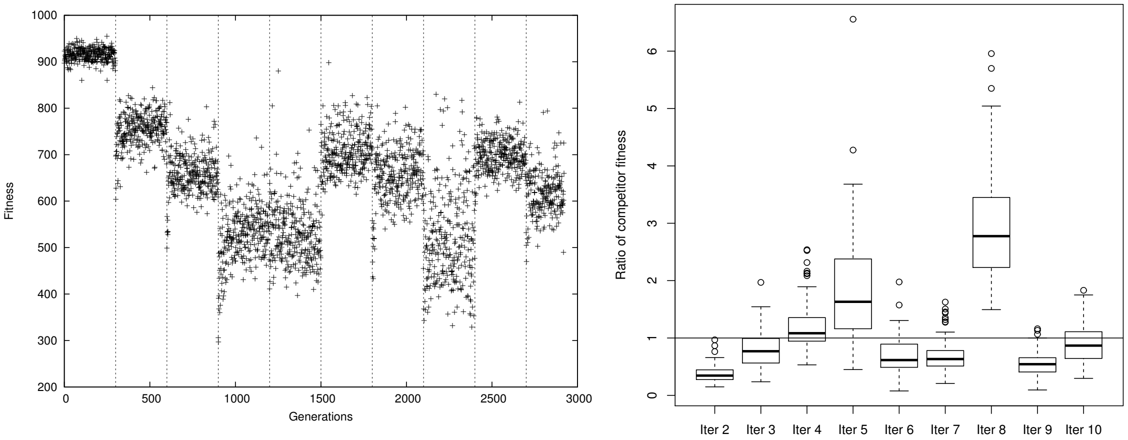


Figure 1: The evolution episodes in the GA. (**left**) The vertical bars indicate the fitness episodes. Initially, there is a fall in fitness. This is followed by a quasi oscillation of high and low fitnesses. (**right**) The ratio of the competitor fitnesses for 100 simulations of the best solution from the GAs. A value < 1 indicates that the evolved solution in the current iteration is better than the solution in the previous iteration. In the second iteration the solutions outcompete the incumbant from the previous iteration. In the third iteration this is also the case, in general. However, for higher iterations competitive solutions fail to evolve, leading to a quasi-oscillation of competitiveness.

sity of *every* reaction after the occurrence of a single reaction and this would be very inefficient. For instance, many reactions of an individual cell would have no impact on the propensity of the reactions in other cells; only those involving the shared external nutrient have an impact on all cells within an environment. Reaction interdependencies are determined from the reactions and rate expressions at the start of a simulation and only the propensities and next reaction times of dependent reactions are updated after a reaction takes place. In addition, a priority queue is used to order the occurrence of reactions, making identification of each next reaction a simple operation.

The software allows a model to be specified in the form of an arbitrary number of reaction equations. The reaction rate may be either a constant or an expression. The specification has separate sections for those elements that are internal species names (e.g. `E1`), evolvable parameter names (e.g. `lk1`), named constants or expressions (e.g. `surfaceArea`). Rate expressions may involve any of these elements. An arbitrary number of species may be designated as shared resources (such as `N1` and `N2`) along with distinctive nutrient 'values' (i.e. energy value to a cell). The evolution of a parameter's value may be either constrained or unconstrained within an arbitrary non-negative floating-point range.

At the start of a simulation, a model is read from a text file and a further efficient feature of the implementation is that the model is converted into a Java class definition that is dynamically compiled at runtime. Concurrency is supported in the software through both multi-threading and the ability to distribute environments across multiple networked machines.

A generation is completed within an environment when any one of the following conditions is fulfilled: all of the cells have died; a time limit has expired; all of the external nutrient has been consumed; no cell within the environment is capable of any further action (i.e., the total propensity of the reaction set is zero). Depending on the requirements of the simulation, when a cell divides there can be a probability of mutation of its parameter values for the daughter cell — as used in the implicit evolution scenario of our model. In addition, crossover is available when a cell is migrated into an environment with an existing population.

Results

For the purpose of this study we made no assumptions about the values of the parameters. Instead, all parameters were evolved using either the explicit or the implicit evolution. We found that the optimisation problem itself is not particularly challenging and the evolutionary algorithms found viable solutions quickly. Indeed, in the implicit evolution the fitness reached close to its maximum value within a few generations.

However, the raw fitness value reached is not necessarily an indicator of the competiveness of the results. A very high fitness solution may perform poorly when competing directly against another solution. This can be illustrated by the sample evolutionary run in Fig. 1 which shows the results of an iterative GA evolution. The first iteration of this simulation produces a solution that, by itself, performs well and reaches a high number of cells. However, when competing against the solution obtained from the second run it will only grow to about one third of its competitor's level. In this sense, the second solution outcompetes the first.

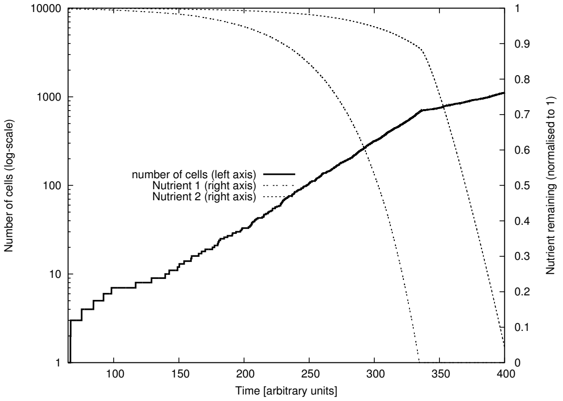


Figure 2: A simulation of a population. The growth of a population over time (left axis) shows a distinct two-phase growth pattern indicated by two straight lines in the logarithmic plot. However, note that there is no lag phase. Initially there is small simultaneous uptake of nutrient. Once nutrient 1 is exhausted the uptake rate of nutrient 2 is increased.

The difference between these two solutions is predominantly the speed with which the nutrient is taken up. This is apparent from the curves labelled "alone" in Fig. 3; these show the two solutions simulated on their own and demonstrate that the first solution grows over much longer time than the second. This is also confirmed by a closer analysis of the simulation data (not shown) and is a feature that is robustly reproduced by repetitive evolutions. Apart from the speed of the uptake, there is also a slight difference in the uptake priority. While the first evolved solution takes up `N2` faster than `N1`, this order is reversed in the simulation of the second solution. Yet, in both solutions there is largely simultaneous uptake for most of the time.

More interesting is the change in behaviour from the second to the third iteration. In this step, there is not only an increase in the speed of uptake, but also a fundamental change in the way nutrient is taken up. In the first two solutions the repression of `P2` by `R` was effectively disabled by a suitable parameter choice. On the other hand, in the third solution the parameters evolved so as to activate this repression mechanism. Specifically, this affected three parameters: the dephosphorylation rate `dR`; the association rate of the repressor and porin 2 `kb`; and the dissociation rate `ukb`. Effective repression can only happen when the dissociation rate is low compared with the association rate. Additionally, repression requires the repressor `R` to be available. If dephosphorilation is too fast compared to the association rate `kb` then effective blocking of porin 2 will not be possible.

The parameters of the third solution evolved so that uptake of the `N2` is strongly reduced while `N1` is taken up. `N2` is taken up at a much higher rate once `N1` runs out. However, note that this is only a partial blocking. There is still a low level of simultaneous uptake of both nutrients (Fig. 2). In this particular series, solutions return to simultaneous uptake of nutrients after the third iteration. In different series of evolutionary runs one would observe a qualitatively similar behaviour, although precisely when the regulated diauxic growth appears will vary between runs. After the first regulated diauxic growth appear, subsequent evolutionary iterations lead to quasi-oscillations between very fit solutions and rather poor ones. Again, this qualitative feature robustly appears in repeated runs (data not shown).

The evolution with genetic algorithms illustrates well how diauxic growth evolves. This can be summarised in three steps: At first, solutions evolve to maximise utilisation of nutrient, irrespective of the relative value of nutrients. Then, in a competitive dynamics, solutions must become faster to prevent competitors from taking up nutrients. At this stage uptake may still be simultaneous. Finally, in a third step solutions evolve sequential uptake of nutrient. This third step depends crucially on the space on the surface of the cells being limited. If there is unlimited space, then only the first two stages will appear and no diauxic growth evolves. This stepwise progression to regulation (and away) can naturally not be observed in the case of implicit evolution because of the nature of this method. Instead, there the solutions evolve directly to regulated diauxic growth.

Anatomy of evolved solutions

In order to understand the adaptive pressures that act on the system, we analysed the populations that we obtained from implicit evolutionary simulations. To do this, we ran the evolution for at least 500 generations and then collected the entire population. From this we removed all unfit solutions — all solutions with fewer than 50 cells in total across all 32 environments. We collected populations in this way over a number of independent evolutionary runs. Altogether, this resulted in 1180 different parameter sets. Fig. 4 shows a boxplot characterising the distribution of parameters across the population. It compares evolutionary runs over two conditions: With limited space for porins and without limited porins.

In the case of a limitation of porins, a strong selective pressure seems to be limited to the leak expression of porins (`leak1` and `leak2`), the parameter controlling the growth rate `KG`, `dR` and `ukb`. The latter two control the regulation of simulateneous growth. In particular, `ukb` is kept very low indicating that there is a strong selection pressure for diauxic growth in the system. The value of `KG` is a Hill constant

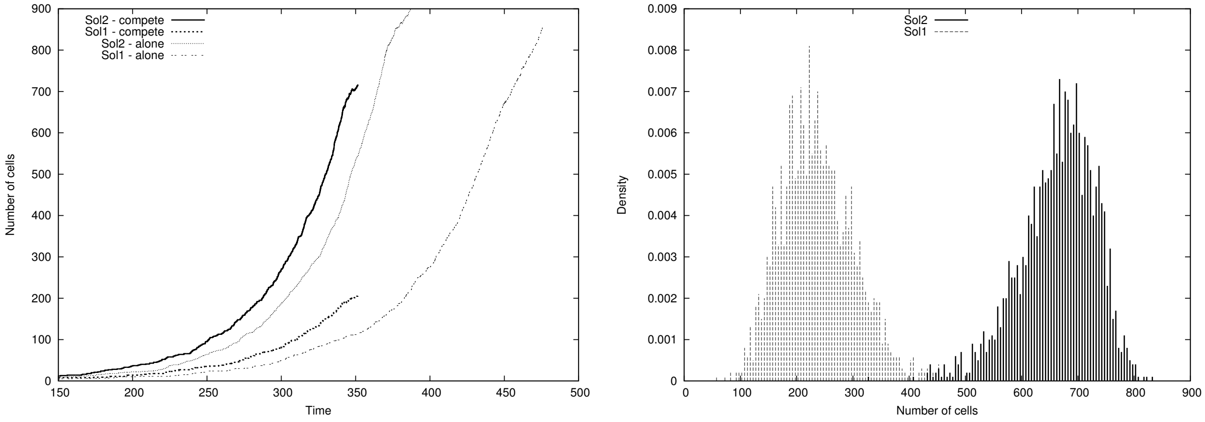


Figure 3: An example simulation comparing the solutions evolved at the first and second iteration (see Fig. 1). (**left**) The curves labelled "compete" show the result of a single simulation where the two solutions are competing for the same nutrient. Clearly, the second solution outcompetes the first one. The graph also shows a simulation of the first and second solution when simulated without competition (the curved labelled "alone"). In this case they grow similar cell numbers. (**right**) A histogram showing the distribution of cell numbers over 2000 competitive runs. There is hardly any overlap between the two distributions. The second solution clearly dominates the first.

and ensures that growth sets in at the correct availability of nutrient.

A notable result is the high leak rates. As can be seen from Table 1 the leak rate determines at what rate porins are expressed even in the absence of an activator. The activation mechanism used here is self-activation in the sense that porin is expressed in response to nutrient presence in the cell, but nutrient can only be present in the cell if there are porins. Hence, a small leak rate is required in order to kick-start the auto-activation of the porin expression system. Yet, a high leak rate essentially usurps the regulation of porin expression in the cell.

In real biological cells, there are two conceivable purposes for regulation of uptake mechanisms. The first is to control over-crowding of the intra-cellular space (or the cell surface for that matter) with proteins that are not needed. The second is to avoid wasting resources on producing proteins that are not needed. Neither is very relevant in the setup of this model for the following reason: our model has a direct negative feedback from surface crowding to porin expression. This means that there is no need for the cell to worry about unnecessary expression. At any time, the cell is either taking up `N1` or `N2`; given the limited surface areas, this means that the negative feedback to expression is always fully engaged and only the porins that are required are actually expressed.

From this it appears that no (primary or direct) regulation of expression is required and therefore there are no disadvantages from a high leak rate. Yet, there are advantages. The main one is that the cell is able to respond quickly to change in circumstances. When `N1` runs out the high leak rate of `P2` makes it possible for production of porin 2 to start without delay. Hence, no lag phase evolves in these simulations.

The limitation of porins is the main rate limiting step in the growth dynamics of the cell. Hence, the solutions that are evolved without this limit show a fundamentally different pattern. When there is no limit on how many porins there can be on the cell surface, i.e. no limit on the maximum rate of importing nutrients, then the rate-limitation moves to the conversion rate of nutrient to cell energy, i.e. `lk1` and `lk2`. Fig. 4 confirms that there is a strong selection pressure on these parameters. In nearly all members of the population they take the maximum allowed value.

On the other hand, the `ukb` and `dR` that were strongly selected in the case of limited surface area are now no longer under this pressure. When there is no limit on the surface area then it would be detrimental to block porin 2. Hence, the repression parameters need to be tuned so as to minimise the binding of `R` with `P2`. The precise values of the relevant parameters are not as important as the fact that the binding rate `kb` is low compared to the unbinding rate and that there is little `R` in the system, i.e. the value of `dR` must not be too low.

Relationship to real parameters

In our simulations, diauxic growth evolves robustly and reproducibly. However, there are crucial differences between the solutions that evolved and the real-world cells. Firstly, there is a small amount of simultaneous uptake of both nutrients. Secondly, there is no lag phase evolving. These two observations are connected but they are also significant with respect to the understanding of the evolution of diauxic growth.

The main parameter governing the length of the lag phase

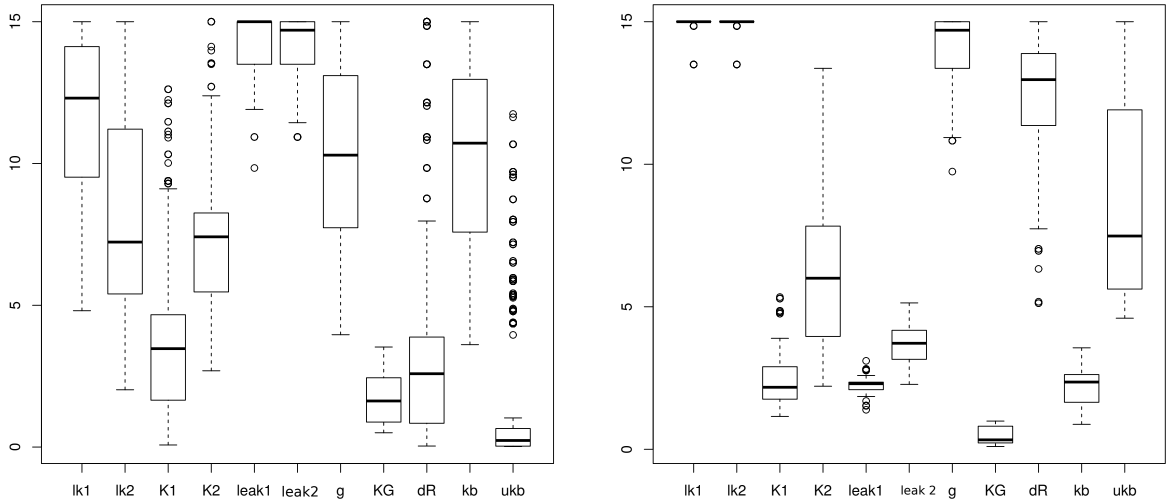


Figure 4: Comparing the parameters evolved. The left plot presents the parameters for the scenario where space on the surface is limited. The right plot indicates a scenario with unlimited space. The clear difference is the parameter `ukb` that evolved to very low values in the case of limited space.

(and the degree of simultaneous uptake) is `ukb`. In order to understand its influence better, we took an evolved solution and compared it with two variants. One where *ukb* was set to 15 — the maximum value permitted within our model — and one where it was set to zero. The former case corresponds to no diauxic growth at all and the latter to a very strong repression of `N2` uptake in the presence of the first nutrient. We confirmed in simulation that with this choice of parameters uptake is entirely sequential, i.e. no `N2` is used before `N1` is exhausted. This also leads to a clearly distinuishable lag phase in the growth curve. The latter case supressed diauxic growth completely and both nutrients are taken up simultaneously. Both of those variants are outperformed by the evolved solution (Fig. 5). This suggests that there is an optimal level of repression of the `P2` metabolism in our model.

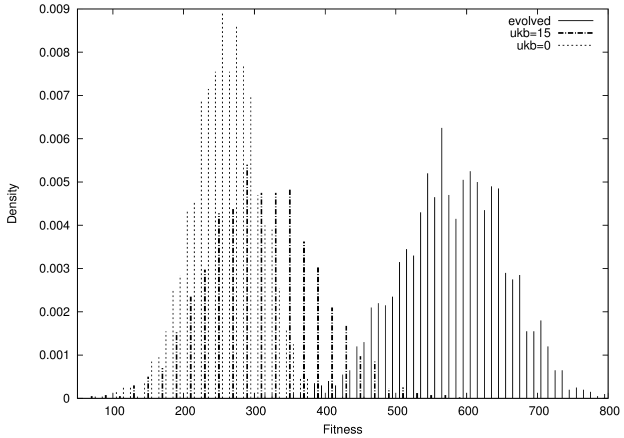


Figure 5: Fitness depends on the parameter `ukb`. The evolved solution shows a distribution of fitnesses. When the unbinding rate is set to zero or to very high values then the fitness goes down substantially compared to the evolved solution. This histograms were produced from 2000 simulations of the model.

Discussion

One possible objection to our simulations is that they are highly simplified with respect to real systems, and hence not comparable. It could be argued that any discrepancy between the evolved solutions and real cells is as irrelevant as any similarities would be coincidental.

Clearly, it is the case that the "bio-chemistry" used here is idealised with respect to the rather more complex web of interactions that regulate diauxic growth in real organisms. However, the complexity of the regulatory mechanisms is not the point. At the heart of the matter is the competitive nature of growing cells and the basic strategy. It is reasonable to assume that cells can adjust the length of the lag phase over evolutionary times. Sequential uptake of nutrients is a positively controlled evolved feature that cells could lose again. So, conceivably, they could also reduce the length of the lag phase or even the strict sequential nature of nutrient uptake. This is confirmed by recent experimental findings that demonstrate this plasticity using artificial wet-lab evolution (Molenaar et al., 2009). For as long as the plasticity of dual nutrient uptake is established, then the details of the implementation are mostly irrelevant. This means that it makes sense to try to understand diauxic growth as a general strategy and it is not necessary to take into account the fine details of the bio-chemical implementation.

The question now is: Why is there an exclusive switch evolving in real cells but the model predicts a soft transition between the two nutrients, with partially overlapping nutrient uptake? One could conjecture that there are delays in the activation of gene expression. Clearly, it takes some time to transcribe and translate genes. This delay in gene production is not represented in the model but, one could suspect, might be responsible for the lag phase during the switch between `N1` and `N2`.

While it is true that there is a minimum time for metabolic pathways to be activated, we think that this is at most a partial explantion for the lag phase. For one, the experimental evidence cited above suggests that lag phases are changeable, suggesting that there are some circumstances in which it is beneficial to have longer lag phases and others when it is better to have shorter ones. This implies that the fundamental limits on gene activation are not relevant. Yet, even in the absence of this evidence, the time required to switch on the genetic machinery does not matter for the question we ask. If a lag phase is detrimental to the fitness of the individual, then the cell could always avoid it by evolving simultaneous, or partially simultaneous utilisation of nutrients. The question remains: Why does it not do that?

In order to understand why lag phases are so common in bacteria, it is necessary to look for fitness trade-offs: something that compensates the cell for the lost growth in the sense that the total growth, compared to competitors, must be higher taking into account both growth and lag phases. This would include growth in different environments. For example, a typical *E.coli* cell will spend much time in environments that do not contain, say, lactose. In those environments, the cell will still express a certain amount of lactose porins by leak expression. This is wasted energy for as long as the cell does not actually get into lactose environments. Hence, if the cell never sees lactose, there will be an evolutionary pressure to adjust leak expression downwards.

On the other hand, if a cell is placed in a lactose environment, then it will be well served having a high leak rate. If leak expression of the lactose uptake system is efficiently repressed, then (on average) it will take a very long time to sense it and switch it back on. During this particular growth episode, this will lead to a long induction period. Or, if taken up in combination with glucose, there will be a long lag phase between glucose and lactose usage. If this scenario were true, a long lag phase would be a side effect of adaptation to a wide range of growth conditions.

Another hypothesis, not necessarily independent of the previous one, is that the lag phase is a consequence of computational cost. The switch from one nutrient to another can be considered as a computation (Zabet and Chu, 2010). Fast and accurate switching comes at energetic cost (Chu et al., 2011) in the sense that maintaining a gene regulatory network that can quickly change from one state to the next and does so accurately, comes at a metabolic cost. The cost of maintaining a highly-performing switch has to be traded off against the gain of fast switching during diauxic growth. This scenario, if true, would entail that the computational/metabolic cost of sensing external nutrients fast enough is higher than the fitness gains from switching fast.

At present, this remains an open question. Addressing this remains a challenge for future work and requires integrative models of nutrient uptake taking into account the entire life-time of the cell. The significance of this question goes well beyond just uncovering a particular microbiological effects. Diauxic growth is fundamentally about cellular and molecular computing. In order to switch its state from metabolising one nutrient to metabolising another one, the cell has to sense external conditions and process this information. At the same time as it senses, it also has to ensure that the process of information processing is economical and properly counter-balanced by the gains of fast switching. As such, diauxic growth makes an interesting case study in biomolecular information processing that integrates questions from evolution, molecular computing and biology.

References

Boianelli, A., Bidossi, A., Gualdi, L., Mulas, L., Mocenni, C., Pozzi, G., Vicino, A., and Oggioni, M. (2012). A nonlinear deterministic model for regulation of diauxic lag on cellobiose by the pneumococcal multidomain transcriptional regulator celr. *PLoS One*, 7(10):e47393.

Boulineau, S., Tostevin, F., Kiviet, D., ten Wolde, P., Nghe, P., and Tans, S. (2013). Single-cell dynamics reveals sustained growth during diauxic shifts. *PLoS One*, 8(4):e61686.

Brückner, R. and Titgemeyer, F. (2002). Carbon catabolite repression in bacteria: choice of the carbon source and autoregulatory limitation of sugar utilization. *FEMS Microbiology Letters*, 209(2):141 – 148.

Chu, D., Zabet, N., and Hone, A. (2011). Optimal parameter settings for information processing in gene regulatory networks. *BioSystems*, 104:99–108.

Deutscher, J. (2008). The mechanisms of carbon catabolite repression in bacteria. *Current Opinion in Microbiology*, 11(2):87 – 93.

Gibson, M. and Bruck, J. (1998). An efficient algorithm for generating trajectories of stochastic gene regulation reactions. Technical Report CaltechPARADISE:1998.ETR026, California Institute of Technology.

Gillespie, D. (1977). Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361.

Molenaar, D., van Berlo, R., de Ridder, D., and Teusink, B. (2009). Shifts in growth strategies reflect tradeoffs in cellular economics. *Molecular Systems Biology*, 5:323.

Monod, J. (1949). The growth of bacterial cultures. *Annual Review in Microbiology*, 3:371–349.

Stülke, J. and Hillen, W. (1999). Carbon catabolite repression in bacteria. *Current Opinion in Microbiology*, 2(2):195 – 201.

Zabet, N. and Chu, D. (2010). Computational limits to binary genes. *Journal of the Royal Society Interface*, 7(47):945–954.

Emergence-focused design in complex system simulation

Chris Marriott[1] and Jobran Chebib[2]

[1]University of Washington
[2]University of Zürich

Abstract

Emergence is a phenomenon taken for granted in science but also still not well understood. We have developed a model of artificial genetic evolution intended to allow for emergence on genetic, population and social levels. We present the details of the current state of our environment, agent, and reproductive models. In developing our models we have relied on a principle of using non-linear systems to model as many systems as possible including mutation and recombination, gene-environment interaction, agent metabolism, agent survival, resource gathering and sexual reproduction. In this paper we review the genetic dynamics that have emerged in our system including genotype-phenotype divergence, genetic drift, pseudogenes, and gene duplication. We conclude that emergence-focused design in complex system simulation is necessary to reproduce the multilevel emergence seen in the natural world.

Introduction

Many of life's mysteries are systems resulting from the interactions between heterogeneous components in a lengthy flow of behavior, yet the results can be predicted due to linear interactions. A mechanical clock is a great example and Rube Goldberg perfected this idea in art with his famous machines (Wolfe and Goldberg, 2000). Such systems may be *complicated* (i.e. have many different components) but they are not *complex* (i.e. system behavior cannot be explained in terms of the behavior of the system's sub-systems).

Other mysteries of life are systems resulting from the interactions between homogeneous or heterogeneous components interacting such that the overall behavior is more than just the sum of its parts (Gleick, 1987; Johnson, 2002). Such systems are called complex or non-linear. Herding is an example of a complex population level behavior emerging from the simple behaviors of homogeneous agents (Reynolds, 1987; Inada, 2001; Olfati-Saber, 2006). From this example and many like it in other domains, complex systems science has itself emerged as we have begun to understand the nature of emergence.

Emergence, as it is currently understood, plays a privileged role in the current organization of science (Morowitz, 2002; Ellis, 2006). As Fodor (1974) famously argued, while we commonly think of psychology, biology, chemistry and the like to be reducible to physics (technically the respective subjects of these studies), it turns out that no reduction exists, or even could exist. From this perspective the objects of one science, like molecules in chemistry, are distinct kinds of entities from another, say atoms and subatomic particles in physics, and the behavior of one cannot be explained simply in terms of the behavior of the others.

Kim (1992) helped bring some metaphsyical peace to reductionism by helping to explain how on the one hand everything *is* physics, but on the other hand no clear reduction exists from one level of science to others. The concept introduced was *supervenience* the idea that the higher order objects and processes were the function of the organization of the underlying objects and processes. Supervenience allowed for a continued commitment to a metaphysical physicalism about science. However it raised an important question, namely, how did these higher-level objects come to be out of the lower-level ones? Today this process is called *emergence* (we refer to what is sometimes called *weak emergence* (Chalmers, 2006)).

If this story is accurate, then the world of chemistry emerged out of physics, and the world of biology emerged out of chemistry sometime later, and now the worlds of psychology, culture, economics, politics, art, and many others have emerged out of biology. The objects of each level organize themselves out of the interacting elements at lower levels. Dennett (1995) calls this process of increasing complexity ratcheting (here we use the term scaffolding). While the examples above have focused on broad levels of emergence, Szathmary and Maynard Smith (2004) argue this type of emergence has occurred many times within biological evolution.

Analytical science has made some progress attempting to handle emergence. This is especially true in the field of evolution. Many analytical models have been developed to predict evolutionary trajectories and their accuracies have been successfully verified in simulation (Hansen and Wagner, 2001; Guillaume and Otto, 2012; Melo and Marroig, 2015). Although analytic models built on understanding evolution in morphometric space may be useful in answering some specific questions, it is not clear that they are still predictive for multi-level complex systems, where fitness landscapes may be non-continuous (Polly, 2008), dynamic, or dependent on the phenotypic distributions of the populations of interest (Nowak and Sigmund, 2004). Thus a primary tool for studying the dynamics of complex systems in evolution and elsewhere is computational simulation (Conrad and Rizki, 1989; Miller et al., 2008; Heath et al., 2009).

Modern simulations of complex systems have led to many breakthroughs in understanding the mechanisms of emergence. However in most of these simulations the same simple methodology is used. In simulation the phenomena of study is treated as emergent, so the underlying phenomena is modeled as a system of interacting parts (Epstein, 1999). The problem is that for simplicity the underlying system itself is commonly simulated as a linear system (simulations of non-linear dynamics all the way down to the physical level are not computationally feasible). However, what is observed in nature is layers and layers of complex emergent systems.

It is our desire to study emergence of complex adaptive systems at multiple levels and in particular we are interested in the co-evolution of culture and biology in humans and other organisms. We believe that one necessary next step in complex system simulation is to incorporate more complicated (heterogeneous, multi-level) component systems and have these component systems emerge from non-linear dynamics whenever possible. We call this *emergence-focused design*.

For our purposes we want to study the emergence of culture. Commonly agent-based models choose to model agents in culture as simple linear systems (Reynolds (1987); Bonabeau (2002); Macy and Willer (2002)). Our desire is to start at least one level lower so we begin with agents with behaviors determined by rich genetic-phenotypic interactions. Our first attempt at multi-level emergence showed promise (Marriott and Chebib, 2014) and so we have extended our model to include a structured environment. This is our first step towards multi-level emergence (higher order emergence from a self-organized, emergent system) and we present the current state of our model.

Model

Our model has incorporated several properties that enrich the dynamics of evolutionary simulation. First, our agents exist in a *structured environment* following a principle that complexity in an agent must reflect complexity in its environmental interaction. Our agents live in a random geometric network of foraging sites. Sexual reproduction requires two agents to be in the same place at the same time which creates a complex social interaction problem.

Second, our agents have a *structured genome* following a principle that genetic dynamics emerge from the physical interaction of an actual genetic sequence. Each agent has a genome represented by a path of foraging attempts at sites in the network.

Third, the genome must interact with the environment to express a phenotype following a principle of decoupling the phenotype from the genotype (a.k.a. ontogenetic development and learning). The path of foraging sites in the agent's genome together with the current site of the agent determine the agent's daily behavior.

Fourth, the agent's survival and reproductive fecundity are functions of their interaction with their environment following a principle of *natural selection* as opposed to artificial selection by a fitness function. A day's foraging costs the agent energy and this energy is replenished by consuming the resources gathered. This constitutes a simple *metabolism* for each agent. If an agent has a daily deficit of energy, it will eventually lose all of its energy and die. If an agent has a daily surplus of energy, then it will store any extra energy over the daily maximum until it can reproduce.

Each of these principles is intended to take a component of the evolutionary system that is often modeled as a linear system and allow it instead to have the potential for emergent dynamics.

Environment

A random geometric network is created by randomly generating n points on a plane and then connecting the close points (Penrose, 2003; Antonioni et al., 2014). For our network we generated points in the range $[0, 1) \times [0, 1)$ and connected points that had a Euclidean distance of at most 0.2. Figure 1 shows a typical map generated in this method. With these settings we usually get a connected network.

Each site in the network has a foraging task for gathering resources. We extend the model of Marriott and Chebib (2014). Each site will have a fixed resource reward for any agent that forages at the site. For typical settings our sites may have a reward of one, two or three resources. Every agent receives the same reward at a site unless the site is depleted for the day in which case the foraging agent receives no reward. Sites become depleted after a fixed number of foraging attempts at the site (we call this parameter the site's yield).

Foraging at a site will cost an agent energy. Energy in our model is temporal energy so corresponds to an amount of time spent on activities. The energy cost is determined by the degree of success of the agent's foraging strategy at the foraging task. The foraging task is to find the rewards that are hidden in five possible locations and the foraging strategy is the order in which hidden locations are inspected. For example, a site with three rewards assumes these three rewards are hidden among five different locations (the number of hiding locations and the number of rewards are parameters that can be varied). The agent must select an order to inspect the hidden locations and each check takes one unit

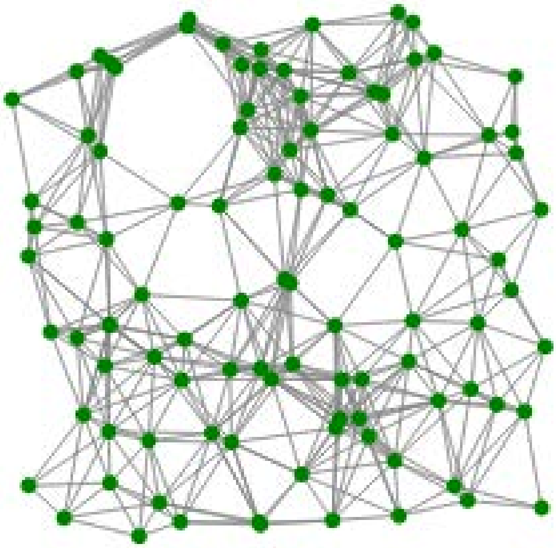


Figure 1: A typical random geometric network with $n = 100$ sites and a connection reach of 0.2.

of energy. So the agent can find the three rewards after three tries, after four tries or after five tries. The minimum energy cost at any site will be one unit of energy per resource gained. The maximum will be the number of hiding locations. The task at a particular site is currently held static throughout the simulation so as to be the target of optimization. Mutations to strategies plus selection will allow for agents to adapt to the foraging tasks.

Moving in the environment also costs the agent energy. The cost is proportional to the length of the edge they travel to get to the new site. In our simulation agents are not allowed to forage at the same site twice in a row, though they may return if they forage at at least one other site first. This is because their foraging behaviors are always represented as a *path* in the network.

The total energy that an agent may spend in a day cannot exceed its current energy reserves or the maximum daily energy expenditure. This means that agents with less than the maximum energy are free to use all of their energy to gather resources. Agents with more than the maximum energy can only spend up to this maximum but the remaining energy can only be stored for reproductive purposes. Energy is replenished at the end of the day by consuming the resources gathered. Each resource consumed generates two energy for the agent (this rate can be varied). In addition to foraging, breeding and travel energy costs, an agent is also penalized daily based on its age. This penalty is a quadratic function of age growing faster as the agent gets older. This ensures agents will eventually die even if they are well adapted to their environment.

At the moment we only test static environments so these features and parameters do not change over the course of the simulation. However, because of the depleted resource sites the agents do experience some minor variation in their environment over the course of the day and between days.

Agent

The agents in our simulation have their behaviors determined by the interaction of their genomes with the environment. The genome of an agent is represented by a path of foraging attempts at sites in the network. This path in the network is a single unbroken path though it may and often does loop back on itself repeating sites. We consider each site in this path a gene in the agent's genome.

A seed agent for our simulation has a randomly generated genome. We build the path by simulating a random walker on the network starting at a random site (Lovász, 1993). At each site we must select a random foraging strategy for the agent to use. This means selecting an order in which the agent will search the hidden locations at the foraging site. We generate a random strategy for the site by selecting a permutation of the five hiding locations uniformly. The gene for this site includes two other components in addition to the foraging strategy. One component determines whether an agent will attempt to breed at the site, and if so, how much energy will be expended in this attempt. This component is randomly generated for the seed agent. Second, the agent must then travel to the next site, and the gene will have an energy cost associated with this travel cost. This cost is fixed by the environment. In this way we can see each gene, if acted upon, will have a total energy cost to the agent equal to the foraging cost plus the breeding cost plus the travel cost. We can then use this metric to measure the energy cost of subpaths of the genome or of the genome as a whole. When randomly generating a genome for a seed agent we generate one with total length about four times greater than the maximum daily energy expenditure. With typical settings this is on average 54 genes.

Once a seed agent is generated we select a gene from its genome at random and start the agent in the site indicated by that gene. Then the agent must select its actions for the day. An agent selects its daily activities by searching its genome forwards and backwards for a subpath beginning at its current site and that maximizes its expected resource gain. The length of the subpath must be less than or equal to the energy the agent has to spend on foraging this day. The optimal path that is found then becomes the behavior of the agent for that day. Since it searches both backwards and forwards (and allow subpaths to rebound off the ends of the genome) there is almost always more than one possible course of action for a day. The agent will always select the optimal option. Also since it can search backwards and forwards, the agent can always follow the optimal path it found yesterday again today, but in reverse order. In this way the agents typically find locally optimal subpaths of their genome to repeat day after day. This subpath is locally optimal, but may not represent the globally optimal subpath in the genome.

Reproduction

An agent can reproduce if it has stored enough energy. How much energy is necessary is determined by what type of reproduction the agent engages in. First, the total cost of reproduction is equal to the daily maximum energy expenditure. This is because the parent agent(s) create a new agent with maximum energy and so observing a principle of conservation of energy the parent agent(s) must transfer that energy from their storage. This means asexual reproduction is twice as expensive to the parent as sexual reproduction, though in both cases the offspring costs the same overall. Agents will only engage in reproduction if their energy bank is enough to afford the cost of reproduction and leave them with at least the maximum daily energy expenditure. With these energy dynamics it means that asexual reproduction is much rarer than sexual reproduction in cases when agents can find sexual partners. In some simulations, asexual reproduction is disabled, though it is necessary when simulations begin with a single seed agent.

Sexual reproduction is carried out between two agents. For this to occur they must both be spending time seeking a mate at the same foraging site for overlapping time periods during the day. This may seem a rare occurrence but due to forces of common descent and convergence the agents develop social organizations like herding, natal philopatry, and assortative mating, and so have no problem finding a mate in our simulations (Marriott and Chebib, 2015).

When a new agent is created from reproduction, its new genome may mutate, recombine with itself, or recombine with another genome (in the case of sexual reproduction). The mutation rate is set at five percent but can be altered. In the simulation there are three kinds of random alterations of the genome.

First, each gene has a mutation probability. Specifically, the foraging strategy in the gene (the permutation) is allowed to mutate. A permutation mutates by swapping two of its entries. This may change the energy cost of that gene. We consider each possible strategy at the same site a different allele of the same gene. This type of mutation is the smallest type of change and it is not used in our current measurements of genetic difference.

Second, the path at either end of the genome may grow with some probability. When this occurs, a random walker is simulated as described for the seed agent above. The random walk begins at the site indicated by the last (or first) gene in the genome. The length of growth is bound by a constant (this is a simulation parameter usually set to 20 genes).

Third, the path at either end of the genome may shrink with some probability. This is done by selecting a length and an end at random and deleting that many genes from the path (the length of the deletion is bound by the same parameter as growth of a genome).

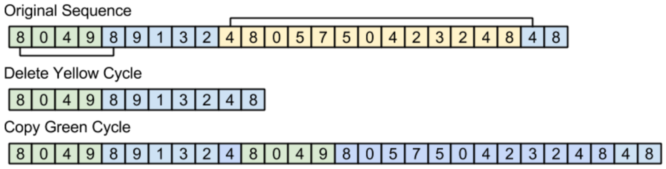


Figure 2: Recombination mechanisms of copying or deleting a cycle. Two possible cycles are selected from the original sequence. The yellow cycle is deleted in the first recombination and the green sequence is copied into a location it fits in the second recombination.

Recombination, another type of mutation that is less random, also has three forms: cycle copy, cycle deletion, and sexual recombination. When an agent mutates it has equal chance of a random growth or deletion (as above) or a cycle copy or cycle deletion. That is, when a mutation occurs, one of a random growth, a random deletion, a cycle copy or a cycle deletion occurs with uniform probability. Thus, half the time the genome gets longer and half the time it gets shorter. Half the time this is due to random mutation and half the time it is due to recombination.

When recombination occurs all cycles in the genome are found (this can be done in a single pass over the genome). A cycle occurs when the path returns to a site it has been to before. Assuming at least one cycle exists (there are often many) we select one uniformly. If a cycle deletion has been selected we delete the cycle. If a cycle copy has been selected we find all suitable locations to insert the cycle and select one uniformly. We then add the cycle to that location. A cycle that starts and ends at site i can replace any occurrence of site i in the genome. See Figure 2 for examples of the recombination mechanisms. It is important to note that there is no maximum cycle length enforced in our simulation at this time. This means the cycle copy can result in a very large copy (in the most extreme case this doubles the size of the genome) or a very large deletion (in the most extreme case this deletes all but one gene). When the whole genome is copied this is called genome duplication (Maere et al., 2005).

During sexual reproduction we also have the likelihood of sexual recombination. The recombination mechanism first computes the maximum number of crossover positions between the two paths by computing the longest common subsequence (similar to synapsis). Every entry in the longest common subsequence is a site that the two sequences have in common. We use these sites as crossover locations when creating an offspring genome. You can imagine lining up the genomes so these sites are next to each other and copy-

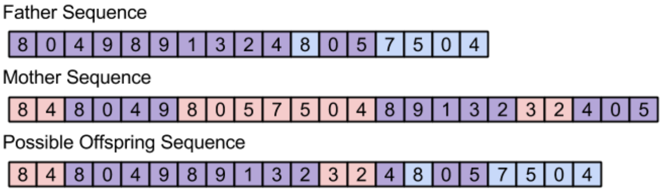


Figure 3: Sexual recombination example. The purple genes indicate the genes on the longest common subsequence. The offspring sequence always includes the longest common subsequence as well as randomly selected adjoining sequences from its parents.

ing the adjoining sequences from either the mother or father sequence.

More formally the offspring genome is created first by selecting one of the two parents to begin copying from. We copy genes from this agent to the offspring agent until we reach the first crossover location. Whenever we reach a crossover location we flip a coin to determine which parent to read from next. We then copy the section of genes from the newly selected agent until we again reach the next crossover location or the end of the genome. We repeat until there are no more crossover locations. Even when there is a lot of overlap between genomes (see Figure 3 as an example) the genes of each parent may have a different allele (i.e. a different strategy at that site). Therefore, which parent we copy the crossover genes from also affects the creation of the offspring's genome.

After carrying out sexual recombination the new genome has probabilities for mutation or recombination with itself as well. A new agent is born on the next day into the site where the parents conceived it. There is the chance that a mutation or recombination has lead to a deletion that left the agent with no strategies given its birth location. When this happens the agent is considered nonviable and dies upon birth.

Observations

Our agent-based model results in very rich emergent genetic dynamics and social structure. This includes the evidence for genotype-phenotype divergence (e.g. phenotypic plasticity), non-coding regions of the genome (Hardison, 2000), pseudogenes (Mighell et al., 2000), genetic drift (Hartl and Clark, 1997), gene and genome duplication (Maere et al., 2005), and degrees of robustness and evolvability of the genome (Wagner, 2012). There are also interesting population level dynamics emergent in our simulation. These include the social organizations of herding (Reynolds, 1987), natal philopatry and assortative mating (Jiang et al., 2013) that emerge around sexual reproduction reported in Marriott and Chebib (2015) as well as population migration, and sympatric/allopatric specialization (Via, 2001). We do not have speciation explicitly built into our model, though genetic divergence of populations is observed even in overlapping territories.

Analytically, we can argue how these features have emerged in our model. For example, consider the divergence of the phenotype from the genotype. One dimension of divergence is phenotypic plasticity, the ability to alter the phenotype in response to environmental features. Two agents in our model that have an identical genotype may occupy different locations in the environment and thus select different phenotypes for the day. The other dimension of divergence occurs as phenotypically similar populations drift apart genotypically. Two agents in our model with identical phenotypes must have some genetic similarity since the overlapping sequence must occur in both of their genomes. However, this need be the only genetic similarities between agents. When additions or deletions occur in regions of the agent's genome that are not used day-to-day these mutations will not affect the phenotype but will contribute to genetic differences. Overtime these can accumulate into large genetic differences between agents that have the same phenotype (see below).

Evidence for these behaviors can be gathered based on a few advantages of our model. First since genomes in our model are sequences we can use standard genome comparison methods like the Levenshtein edit-distance (Levenshtein, 1966) to calculate mutation distance between agents. A relevant phenotype of our agents is the sequence of sites they visit over the day. Since the phenotypes of the agents are also sequences, they can be compared in the same way. Figure 4 shows one population of 159 agents after evolving for 10000 days. In the figure we use four heat maps to represent the variation in genotypes and phenotypes. In each heat map the the rows and columns represent agents (in the same order) and the intersections have a color indicating how different the two agents are (genotypically or phenotypically). We use single link hierarchical clustering to group our agents together based on their phenotypes and genotypes. We see that when we cluster agents based on phenotype we get collections of agents we call herds in Marriott and Chebib (2015). These can be identified as dark squares along the diagonal of the top right heat map in Figure 4. One herd has been indicated in purple. When we look at the same group of agents' genotypes (top left heat map) we find that though there is some similarity, there are clearly at least two different genetic groups in this herd. When we cluster the agents based on genotype instead (bottom left heat map) we see three clear genetic groups. One is indicated in purple. It is not hard to see that these groups consist of agents with widely different phenotypes (bottom right heat map).

We know that the regions of DNA serve different roles roughly grouped as coding and non-coding regions. Coding regions code for functional proteins whereas non-coding re-

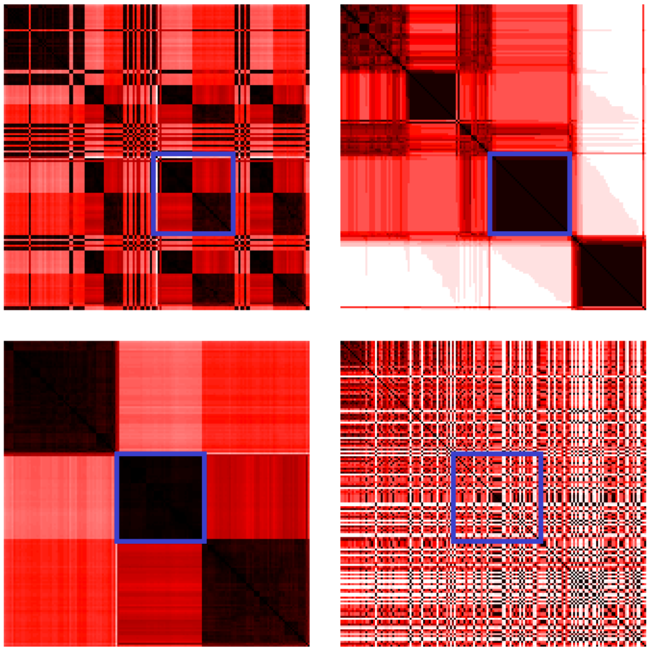


Figure 4: Genotype-phenotype divergence. Genotype (left column) and phenotype (right column) difference heat maps for an evolved population of 159 agents. Black indicates similarity, white difference, and shades of red in between. Agents are clustered by phenotype (top row) or genotype (bottom row). The region highlighted in purple in the top row shows a phenotypically similar herd consisting of agents with different genotypes. The region highlighted in purple in the bottom row shows a population of genetically similar agents with different phenotypes.

gions range from coding for other functional elements like RNA to a catch all "junk DNA" for regions that have no known function. Some non-coding sequences called *pseudogenes* are either genes that have lost their ability to be expressed or have otherwise ceased expression. The identifiers of pseudogenes are their homology to other genes and their lack of expression. Applying this definition to our model we can suggest that the regions of the genome that are not used by the agents are also homologous to regions that are used at least in so far as they potentially encode a phenotype. So for this discussion we'll consider the region of the genome used by the agents day-to-day to be the *expressed gene sequence* and the region that is not used to be the *psuedogene sequence*.

An important factor of the pseudogene sequences in our model is that they are neutral to selection but have potential functionality or even sub-functionality (Qian et al., 2010; Guillaume and Otto, 2012). Mutations in the pseudogene sequence are commonly neutral because these regions are not used. Assuming that mutations occur at a steady rate, then these neutral mutations can accumulate. This process is called genetic drift. Since the expressed gene sequence is rarely more than twenty genes in our model we can see genetic drift in the large pseudogene sequence.

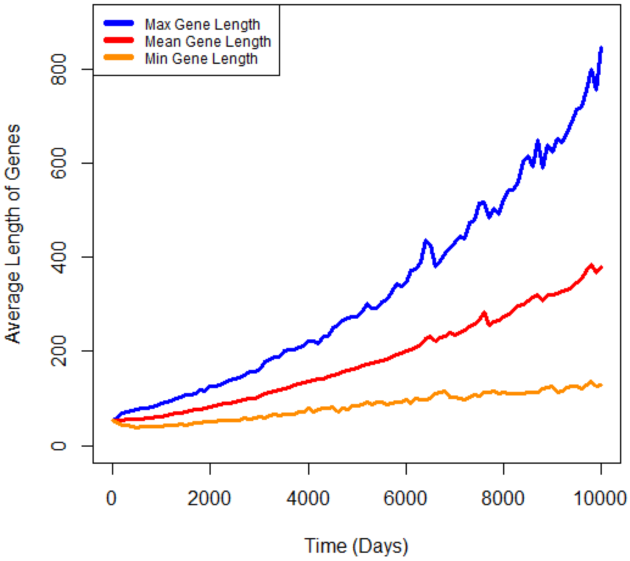


Figure 5: Genome length over time. Maximum, mean and minimum genome length averaged over hundreds of independent runs.

This drift occurs in two ways. First, the alleles in the pseudogene sequence will mutate (though these changes are not reflected in the phenotype and thus are not selected for or against). As a result there is no trend towards optimization in the alleles of the pseudogene sequence, whereas we do see such optimization in the expressed gene sequence. Second, additions and deletions in the psudogene sequence also have no effect on the phenotype. As a result the length of the genome drifts due to these additions and deletions. Recombinant additions (cycle copying) are unbounded in length so this can cause the length of the genome to become quite long over several generations by copying regions of the pseudogene sequence. Figure 5 shows the trends in genome length over time. This data represents the average over hundreds of runs. Both average length and variation in length increase over time. This phenomena maybe caused by genetic drift or gene sequence duplication (see below).

The cycle copying recombination can play a different role when copying a region of the expressed gene sequence of the genome. All cycle copying recombination can be seen as a type of gene duplication because the duplication of what we are calling genes (we are copying the specific alleles in these cases). When this occurs in the expressed gene sequence we can also consider this operation a gene sequence duplication. In particular, consider a special case in which the entire expressed gene sequence is duplicated. Assuming this is the first time this recombination occurs in our agent, then it has two identical locally optimal gene sequences. When daily activity selection occurs one of the two identical regions will be selected. The other sequence will be part of the pseudogene sequence because it is not expressed. This will not result in any phenotypic difference between the child agent and the parent agent that had only one such region. Nonetheless, consider the difference allelic (or other) mutation will have on the parent and child. The parent, with only one optimal region, has a high risk attached to mutating the region of use. If the alleles in that region mutate they more often than not become less efficient. Such mutations will likely mean the new agent is less fit. This result will likely have a negative impact on the reproductive success of the agent.

Consider now the child agent with two such regions. Mutation that occurs to one will not affect the other. Thus if a negative allelic mutation occurs on one of the regions but not the other the agent is insulated from this mutation. Its phenotype is unaltered as it can use the non-mutated region. Being insulated from negative mutations is a form of genetic robustness. With this added robustness, the genome of the child agent is now able to explore the allelic space with less risk. The chance of a positive mutation affecting the phenotype has doubled (it can occur in either region) but the chance of a negative mutation affecting the phenotype has halved (it has to occur in both regions). This is a step toward what is called greater evolvability. The gene sequence duplication we describe here results in greater robustness and evolvability. It can also occur more than once (and in our model can potentially increase the number of regions exponentially per duplication). Each duplication further impacts robustness and evolvability.

Extending our Model

The genetic and population level dynamics we have encountered in our model, which may be among others we have not yet identified, are due to the commitment to emergent dynamics in as many aspects of our model as possible. For instance the genetic dynamics discussed here are largely due to evolving a *sequence of genes* with a realistic copying mechanism as is the case with DNA. Even though this mechanism plays a major role, the structured genome could not exist without the structured environment. Further the dynamics are greatly limited if we do not have genotype-phenotype divergence or metabolic systems (Marriott and Chebib, 2014). We have tested a mechanism in which the whole genome is used by the agent (it travels back and forth along the genome) which causes convergence on short optimal genomes but is also less dynamic.

Too often steps are made to simplify the model computationally. This typically means reducing one or many of these genetic sub-systems to linear dynamics. This then removes possibilities for synergistic emergent dynamics among these systems. Even so, there are many aspects of our model that still rely on simple rules and linear systems. A short list of future improvements we would like to make are adding dynamism to the structure of the environment, increasing the genetic and environmental interaction, developing the energetic metabolism of our agents, and adding more genetic copying and deleting mechanics.

The sequential properties of paths in a network allow for natural mechanisms of mutation and recombination to be employed. These in turn lead to dynamics similar to those observed in nature and alow for the application of many biological concepts through analogy to our artificial agents. Our agents make use of a foraging analogy (Mobus, 2000) but evolving paths in a network can be used for other ends including optimization and solving other network-based problems in complex systems. We encourage others to investigate the richness of this model.

In the short term, we plan to explore in more details the nature of the genetic dynamics displayed by our model. The mechanisms of genetic drift and gene duplication deserve further study. At present we are developing means of detecting and tracking events of recombinant duplication. We are also currently developing metrics of robustness and evolvability for our agents that are independent of the gene sequence duplication. We can use these to determine the impact of gene sequence duplication on the robustness and evolvability of the genomes of our agents. Further we have observed allopatric and sympatric specialization of populations in different regions. Simple examples of this (see Marriott and Chebib (2015)) are the result of migration and separation in space (allopatric) or assortative mating (sympatric).

The goal of our long term research is to understand the origins of social behavior and social learning in artificial and biological populations. We feel it is critical to these goals that the manner of social interaction in our models is not scripted as is common in multi-agent systems and agent based models in the social sciences. Instead we desire a model in which social behaviors emerge under selection from underlying non-social forces. Our current model demonstrates this first stage of development.

The next major step is to allow those emergent social behaviors to create a new domain of interaction that can lead to higher level emergence. We call this a cascade of emergence. Complex life and especially the social behavior of humans is due to such a cascade. To accomplish this we are developing coupled emergent systems for individual and social learning. These are modeled off of our earlier work in Marriott and Chebib (2014).

References

Antonioni, A., Bullock, S., and Tomassini, M. (2014). Reds: an energy-constrained spatial social network model. In *The*

Fourteenth Conference on the Synthesis and Simulation of Living Systems, volume 14. MIT Press.

Bonabeau, E. (2002). Agent-based modeling: Methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7280–7287.

Chalmers, D. J. (2006). Strong and weak emergence. *The reemergence of emergence*, pages 244–256.

Conrad, M. and Rizki, M. M. (1989). The artificial worlds approach to emergent evolution. *BioSystems*, 23(2):247–258.

Dennett, D. (1995). *Darwin's dangerous idea*. New York: Simon and Schuster.

Ellis, G. F. (2006). On the nature of emergent reality. *The re-emergence of emergence*, pages 79–107.

Epstein, J. M. (1999). Agent-based computational models and generative social science. *Generative Social Science: Studies in Agent-Based Computational Modeling*, 4(5):4–46.

Fodor, J. A. (1974). Special sciences (or: the disunity of science as a working hypothesis). *Synthese*, 28(2):97–115.

Gleick, J. (1987). *Chaos: Making of a new science*. New York: Penguin.

Guillaume, F. and Otto, S. P. (2012). Gene functional trade-offs and the evolution of pleiotropy. *Genetics*, 192(4):1389–1409.

Hansen, T. F. and Wagner, G. P. (2001). Modeling genetic architecture: a multilinear theory of gene interaction. *Theoretical population biology*, 59(1):61–86.

Hardison, R. C. (2000). Conserved noncoding sequences are reliable guides to regulatory elements. *Trends in Genetics*, 16(9):369–372.

Hartl, D. L. and Clark, A. G. (1997). *Principles of population genetics*. Sinauer associates Sunderland.

Heath, B., Hill, R., and Ciarallo, F. (2009). A survey of agent-based modeling practices (january 1998 to july 2008). *Journal of Artificial Societies and Social Simulation*, 12(4):9.

Inada, Y. (2001). Steering mechanism of fish schools. *Complexity international*, 8:1–9.

Jiang, Y., Bolnick, D. I., and Kirkpatrick, M. (2013). Assortative mating in animals. *The American Naturalist*, 181(6):E125–E138.

Johnson, S. (2002). *Emergence: The connected lives of ants, brains, cities, and software*. Simon and Schuster.

Kim, J. (1992). Multiple realization and the metaphysics of reduction. *Philosophy and Phenomenological Research*, pages 1–26.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Lovász, L. (1993). Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1):1–46.

Macy, M. W. and Willer, R. (2002). From factors to actors: Computational sociology and agent-based modeling. *Annual review of sociology*, pages 143–166.

Maere, S., De Bodt, S., Raes, J., Casneuf, T., Van Montagu, M., Kuiper, M., and Van de Peer, Y. (2005). Modeling gene and genome duplications in eukaryotes. *Proceedings of the National Academy of Sciences of the United States of America*, 102(15):5454–5459.

Marriott, C. and Chebib, J. (2014). The effect of social learning on individual learning and evolution. In *The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 736–743. MIT Press.

Marriott, C. and Chebib, J. (2015). Finding a mate with no social skills. In *Proceedings of the 2015 conference on Genetic and Evolutionary Computation*. ACM.

Melo, D. and Marroig, G. (2015). Directional selection can drive the evolution of modularity in complex traits. *Proceedings of the National Academy of Sciences*, 112(2):470–475.

Mighell, A., Smith, N., Robinson, P., and Markham, A. (2000). Vertebrate pseudogenes. *Febs Letters*, 468(2):109–114.

Miller, J. H., Page, S. E., and LeBaron, B. (2008). *Complex adaptive systems: an introduction to computational models of social life*. Princeton: Princeton University Press.

Mobus, G. (2000). Foraging search: Prototypical intelligence. In *Proceedings of the Conference of the American Institute of Physics*, pages 592–605. Institute of Physics Publishing.

Morowitz, H. J. (2002). *The emergence of everything: How the world became complex*. Oxford University Press.

Nowak, M. A. and Sigmund, K. (2004). Evolutionary dynamics of biological games. *science*, 303(5659):793–799.

Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *Automatic Control, IEEE Transactions on*, 51(3):401–420.

Penrose, M. (2003). *Random geometric graphs*. Oxford University Press Oxford.

Polly, P. D. (2008). Developmental dynamics and g-matrices: Can morphometric spaces be used to model phenotypic evolution? *Evolutionary biology*, 35(2):83–96.

Qian, W. F., Liao, B. Y., Chang, A. Y. F., and Zhang, J. Z. (2010). Maintenance of duplicate genes and their functional redundancy by reduced expression. *Trends Genet.*, 26(10):425–430.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM Siggraph Computer Graphics*, 21(4):25–34.

Szathmary, E. and Maynard Smith, J. (2004). *The major transitions in evolution*. Oxford University Press.

Via, S. (2001). Sympatric speciation in animals: the ugly duckling grows up. *Trends in Ecology & Evolution*, 16(7):381–390.

Wagner, A. (2012). The role of robustness in phenotypic adaptation and innovation. *Proceedings of the Royal Society B: Biological Sciences*, 279(1732):1249–1258.

Wolfe, M. F. and Goldberg, R. (2000). *Rube Goldberg: Inventions!* Simon and Schuster.

The Lattice of Chemical Organisations

Pietro Speroni di Fenizio [1]

[1] Bio Systems Analysis Group, Jena Centre for Bioinformatics, Friedrich Schiller-University Jena, D-07737 Jena
ecal2015@piespe.net

Abstract

The paper describes how, in an Artificial Chemistry under flow conditions, the set of organisations form a lattice. The consequences of this are described, in particular how a series of theorems, valid for lattices, can be applied to more easily discover the complete set of organisations. An algorithm is then developed that uses such theorems to explore such lattice. The algorithm is applied first to the NTop Artificial Chemistry and then to an extension of it. Due to its complexity this system is also suggested as a benchmark case to test new Artificial Chemistries' algorithms.

Introduction

In 1994 Walter Fontana and Leo Buss introduced the concept of Organisation to represent the fixed points in Constructive Dynamical Systems (Fontana and Buss, 1994a, 1994b, 1996). Looking at Dynamical Systems they observed how they could predict only quantitative dynamics, and no qualitative development. True novelty could never appear through Ordinary Differential Equations. But novelty is inherent in this world. And it is unpredictability is what makes the world so interesting. So, to describe how a system could transform the qualitative space of possibilities, they presented the concept of Constructive Dynamical Systems. Those were a molecular based system, where new molecules could be generated through the interaction of existing ones. Such system was thought as a model for the macro-molecules inside a cell, but this was just one of the possible examples, of such a general theory.

Using standard dynamical systems as a metaphor, they then went on describing what would be the equivalent fixed points in their system; points where no novelty would appear, and called such cases Organisations. A set of molecules would be an organisation if, and only if, for each molecule inside the set, there was a reaction among the molecules in the set that would produce it; and, given any reaction among the molecules of the set, the result would always be a molecule of the set. The first property was called Self-Maintenance and the second Closure. A set satisfying both of those properties would be called an Organisation.

Later (Dittrich, Speroni di Fenizio, 2007) Organisations were then renamed Semi-Organisations when it became clear that those properties were not enough to permit to those sets to be dynamically stability. Organisations were so re-defined as special Semi-Organisations where it is possible for all the reactions among molecules inside them to be active, and have no molecule type diminish.

Studying Organisations and Semi-Organisations, it was soon found that under quite common assumptions, those structures would form a Lattice. It should be noted that this it is not always true for every reaction system, but it is true if we assume that every molecule has a certain probability to disappear (i.e. every molecule would have an out-flux greater than zero). Those systems were every molecule has an out flux were called *Flow Systems* to distinguish them from the more general *General Reaction Systems* (Dittrich, Speroni, 2007). It should be noted how all of Fontana and Buss' models had an out-flux applied to each molecule that would destroy it at the same speed at which others were produced. So those models were Flow Systems, and the set of organisations would form a Lattice.

Other Flow Systems are also possible; for example if we consider cells, and thus cell growth, although it is not true that each molecule has a certain probability higher than zero of being excreted, it is true that as the cell grows the relative concentration of each molecule that is not generates diminishes, having the same effective result as if each molecule was subject to an out-flux. Then when the cell reproduces the average amount of that molecule halves, eventually reaching zero after enough reproductions. Thus the set of molecules inside a cell form a Flow System, and the set of possible Organisations in a living system form a lattice. Vice-versa, the reaction system in the atmosphere of a planet is not a Flow System, but a General Reaction System, and the Organisations do not form a lattice but just a Partially Ordered Set. Flow Systems and General Reaction Systems are not the only possible type of Artificial Chemistries: a more detailed analysis can distinguish also Catalytic Flow Systems as a specific type of Flow System where each molecule reacts in a catalytic way. In other words they are not used up in the

Catalytic Flow Systems	**Flow Systems**	**General Reaction Systems**
All molecules in each reaction are catalytic; all molecules have an out flux	*All molecules have an out-flux*	*No requirements on the out-flux nor on which molecules are catalytic*
each semi-organisation is an organisations	some semi-organisations are organisations	some semi-organisations are organisations
both organisations and semi-organisations form a lattice	both organisations and semi-organisations form a lattice	neither organisations nor semi-organisations form a lattice

Table 1: Different type of Artificial Chemistries produce sets of organisations with different properties.

reaction, while the prime material that generates the result is supposedly coming from a substrate of basic material floating around. Substrate to which each molecule eventually decays through the out-flux. Catalytic Flow Systems, Flow Systems, and Reaction Systems are quite different in terms of their relative properties (see Table 1).

This paper aims to investigate the consequences of the fact that under flow conditions, the set of Organisations form a lattice.

What does it mean to *understand* an Artificial Chemistry

In this context understanding an Artificial Chemistry means having a list of all the possible organisations; for each organisation know what are the organisations directly above and below it. With the organisations A being directly above (directly below) the organisation B we mean that A contains B (B contains A), $A > B$, ($A < B$) and there is no other organisation C that contain B and is contained by A (that contain A and is contained by B), $\nexists$ C such that $A > C > B$ ($\nexists$ C such that $A < C < B$). Thus being able to predict, given a starting condition, where the system will likely evolve, and if we mix two different systems what the result of it will be. I am speaking here of broad qualitative prediction, not precise quantitative ones. Knowing what molecules will be present, while ignoring the relative quantities, is a satisfactory result.

Suppose you have two test tubes; each with a different experiment of the same Artificial Chemistry (AC); each having a small out-flux, thus that we can be sure that the set of organisations of this AC is a lattice.

If we consider the molecules inside, eventually they will express an organisation, as only the molecules which can be generated will be present and all the others will be lost through the out-flux. What will happen if we join those two test tubes? Because the organisations in a Flow System form a lattice, we can immediately say that such action will generate the organisation union of the two organisations. And this exist and is unique because in a lattice the union of two elements is always present and unique. Of course if the organisations present in the two test tubes are one inside the other, then the union is trivially the biggest one. More interestingly we can ask: suppose we have two test tubes, where two organisations are expressed, A and B; suppose we want to add some molecules to both A and B without changing the organisation inside; what can we add? And the answer is: we can add anything from the organisation C, where C is the organisation intersection between A and B. Again we know that C uniquely exists because the set of organisations form a lattice.

From this appears obvious that we do not only need to know the set of organisations, but also know the content of the two tables that given any two organisations A and B would tell us their union ($A \cup B$) and their intersection ($A \cap B$). Calculating those tables is also something that becomes more efficient by the fact that those organisations form a lattice and thus we can use theorems from lattice theory to calculate some of the results.

Thus understanding the set of organisations gives us some clear, practical abilities to understand what happens in an Artificial Chemistry. And when this is a lattice, it all becomes easier.

Basic Known Definitions from Previous Work

In a reaction system a closed set C is a set of molecules such that given any two molecules their reaction is still contained in C. A semi-self-maintaining set S is a set such that given any molecule m that is consumed in S, there exist a reaction inside S such that m is produced. Any set that is both semi-self-maintaining and closed is a semi-organisation. A self-maintaining set S is a semi-self-maintaining set such that there exist a reaction speed for all reactions among the molecules in S, with each reaction speed higher than zero, and such that each molecule has a production rate higher or equal to zero. In laymen's terms, a semi-self-maintaining set is a set where all molecules can be produced, and a self-maintaining one is one where this is globally possible. In Catalytic Flow systems the two coincide as every semi-self-maintaining set is self-maintaining [Speroni di Fenizio, 2007].

Given any set F, we can generate its closure as the smallest closed set containing F. This always exists (albeit it can be infinite in size, if the AC contains an infinite diversity of molecules) and it is unique; we will indicate this as $G_C(F)$.

Given two closed sets, we can define the closed union ($\sqcup_C$) as the closed set generate by the union of the two sets. Thus $\forall$A, B, closed sets; $A \sqcup_C B \equiv G_C (A \cup B)$. Similarly we can define, $A \sqcap_C B \equiv G_C (A \cap B)$. And then if C is the set of Closed Sets, $<C, \sqcup_C, \sqcap_C>$ will be a Lattice.

Given any set F we can generate its semi-self-maintenance subset $G_{sSM}(F)$ as the biggest semi-self-maintaining subset contained in F. Again this exists and is unique. In a Flow System (but not in always in a reaction system), if we take a closed set C and we apply the operator $G_{sSM}(C)$ the result is still closed. So given any set F we can associate a semi-organisation to it as $G_{sO} (F) = G_{sSM} (G_C (F))$. This can be expanded by defining the operator Generate Self Maintaining set: G_{SM} (T) as the operator that returns the biggest Self Maintaining Set contained in T.

Applying the equivalent definition of union and intersection among semi-self-maintaining sets, self maintaining sets, semi-organisations, and organisations, we obtain that in Flow Systems the set of semi-self maintaining sets, self maintaining sets, semi-organisations and organisations are all lattices with their respective union and intersection. Thus if O is the set of organisations in a Flow System $<O, \sqcup_O, \sqcap_O>$ is a lattice. But not necessarily in a reaction system. We shall now explore the consequences of this.

Note: In the rest of the paper we shall use $\sqcup$ to mean $\sqcup_O$ and $\sqcap$ to mean $\sqcap_O$.

Useful Theorems

There are several theorems from Lattice theory which can help in mapping the lattice of organisations, and thus understanding better an Artificial Chemistry. Generally speaking if we have two organisations A, B then $A \sqcup_O B = G_{SM} (G_{sSM} (G_C (A \cup B)))$. Those calculations can be long, so if we can shortcut the calculations just by working on the knowledge we already have this speeds up sensibly the work. In passing we note that in a Flow System for all sets A, although $G_{SM} (G_{sSM} (A)) = G_{SM} (A)$, it is usually faster to calculate it as $G_{SM} (G_{sSM} (A))$ as $|G_{sSM} (A)| < |A|$, and G_{sSM} being an algebraic operator is usually faster to calculate than G_{SM}.

Theorem 1: If A, B and C are elements in a Lattice $<L, \cup, \cap>$, then we know that $(A \cup B) \cup C = A \cup (B \cup C)$.
Now suppose that we have two organisations, S, C, and we are interested in calculating $T = S \sqcup C$. If we can express S as $S = A \sqcup B$ then $T = S \sqcup C = (A \sqcup B) \sqcup C = A \sqcup (B \sqcup C)$. And if we know the value of $B \sqcup C$ (for example, $B \sqcup C = R$) then $T = S \sqcup C = A \sqcup R$. Similarly if we need to calculate $A \sqcup R$. Since $R = B \sqcup C$ then $S \sqcup C = A \sqcup R$. So calculating the union of one of those relations will solve also the other. Same reasoning can be applied for the intersection.

Theorem 2: Suppose we have three organisations, A, B and C with A contained into B and B contained into C (thus $A < B < C$). And let R be a fourth organisation. If we want to calculate $B \sqcup R$; and we know that $A \sqcup R = C \sqcup R$ then $B \sqcup R = A \sqcup R$. In other words if we have two organisations, A and B and, when interacting with an external organisation R both will give a particular result S (that is $S = A \sqcup R = C \sqcup R$), then every other organisation in between A and C if united with R will be equal to S.

Theorem 3: Suppose we have two organisations, A, S with $A < S$ then $A \sqcup S = S$ then we can consider that given any other organisations B, such that $A \sqcup B = S$, for every organisation C such that $A < C < S$ then $C \sqcup B = S$. Symmetrically for every D such that $B < D < S$, $A \sqcup D = S$. But since $C \sqcup B = C \sqcup S$, and $B < D < S$ then $C \sqcup D = S$.
So once we calculate a single union we can fill in several entries of the Union Table. The same symmetric reasoning, with exactly the same dual theorem, can be applied for the intersection. We will now use those theorems to find the lattice of all organisations.

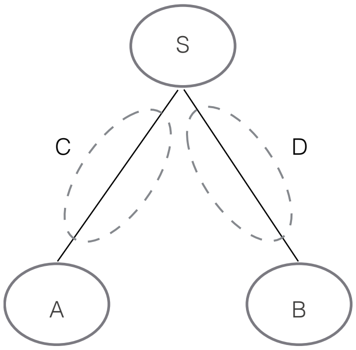


Figure 1: Theorem 3. If $A \sqcup B = S$, any organisation C between A and S united with any organisation D between B and S will always produce S. We do not need to calculate $C \sqcup D$.

Finding the Lattice of Organisations

Let M be a set of Molecules, with a reaction * such that $\forall$ a, b $\in$ M: $a*b \in M \cup \varnothing$. Let <M, *> be an Artificial Chemistry. Let <**O**, $\sqcup$, $\sqcap$> be the lattice of Organisations over <M, *>.
We need to find all the organisations in **O**. Let us start by assuming that we have two basic organisations, the empty set $\varnothing$, and M: $\varnothing$, M $\in$ O. Those will be the top and the bottom organisations in the lattice. Every lattice with a finite number of elements has always a top element (the union of all the elements, also called the 1) and a bottom element (the intersection of all the elements, also called the 0). So we start by taking the bottom element as the organisation generated by the empty set, and the top element as the organisation generated by the set of all molecules.
One obvious solution to find all organisations is to check every single subset of M, which means checking $2^{|M|}$ subsets. If M is big this could be impractical or simply impossible. Let's look at ways in which we can exclude some sets without testing them. In fact without even listing them.
To reach **O** we are going to build a chain of sub-lattices of organisations, $N_0 < N_1 < \ldots < N_m =$ **O**, with N_0 the sub-lattice generated by the empty set and by the set of all molecules: $N_0=<\{G_0(\varnothing),G_O(M)\},\sqcup,\sqcap>$. Note that given a set of organisations P, it is trivial to build a sub-lattice out of it just by taking the closure respect to $\sqcup$ and $\sqcap$.
Thus once we have the sub-lattice N_i, and we find another organisation H, we can calculate N_{i+1} as $G_{C\sqcup\sqcap} (N_i \cup H)$. Where with $G_{C\sqcap\sqcup}$ (P) we indicate the set of all organisations that can be generated by recursively applying the organisation union, $\sqcup$, and the organisation intersection, $\sqcap$, between organisations in P and the organisations generated in this process.
When we are doing this we also store two tables, the Table of Unions of N_i: $T\sqcup_i$, and the Table of Intersections of N_i: $T\sqcap_i$. In general for every A, B in N_i, $T\sqcup_i$ will store $A \sqcup B$ and $T\sqcap_i$ will store $A \sqcap B$. And being N_i a sub-lattice, we know that both $A \sqcup B$, $A \sqcap B \in N_i$. Thus we know that if the tables $T\sqcup_i$ $T\sqcap_i$ are complete with every union and interaction calculated, and the result is an organisation known we do not have to proceed further, and we have found a sub-lattice. It is important that this process of calculating all the organisation union and intersection is made as efficient as possible. In this we are helped by the fact that **O** is a lattice, and thus we can apply the theorems listed above.

Adding one molecule at a time

Let us start by assuming we have a sub-lattice of organisations N_i, then $\forall$ B $\in$ N_i, we need to test all the molecules that are not in B, so $\forall$ *e* $\in$ M \ B, we need to study B*e*[1]. Calculating B*e* leads to 3 possible outcomes:

1) B*e* = B;
2) B*e* = C with e $\notin$ C and B < C;
3) B*e* = C with e $\in$ C and B < C.

In the first case we will say that *e*, in the context of B, goes down (case 1); goes by the side (case 2); or goes up (case 3).
So we go through all the molecules and we store for each molecule *e*, if in the context of B, *e* goes upward, downward, or sideward. We also store the generated organisation B*e*.
For each new organisation H that we find, we expand N_i into N_{i+1} by adding the organisation to the list of known organisations, and calculating the union and intersection closure. Of course, if $B \cup \{e\} = C$, with C already a known organisation, we don't need to test B*e* at all, and we know we are in case 3.

[1]**Note**: In this paper we will indicate briefly $G_O(\{e\}\cup B)$ as B*e*, for every molecule e, and similarly if we need to test a set adding two molecules, e, f, we will indicate $G_O(\{e\}\cup\{f\}\cup B)$ as B*ef.*

cases		*e* goes		
		Upward	Sideward	Downward
f goes	Upward	1	2	3
	Sideward	2	4	5
	Downward	3	5	6

Table 2: Will adding two molecules at the same time to an organisation produce novelty? Depends on what each molecule does by itself. Six cases are possible.

Once we have finished exploring what happens by adding one molecule to a specific organisation, we continue with the next organisation, still maintaining the memory of all the molecules that added to B go up, on the side, or down.

Also note that if B*e* goes sidewards generating C, then C*e* will go downward, so we do not need to test it.

Once we have finished exploring for each known organisations (which included the new ones we might have found in the meantime), what happens when we add one molecule, we need to consider what happens when we add two molecules. This is where some shortcuts will be possible.

Adding more than one molecule at a time

Let us start by saying that if we have two sets of molecules x, y, and two organisations B and C, with B < C, we do not need to test B*x* if we can prove that B*x* = C*y*. Because we will already consider the organisation generated by B*x*, when we consider the organisation generated by C*y*. Let us suppose we take an organisation B, and two molecules *e*, *f* in M\B. Do we need to test B*ef*? We need now to look at the combinations depending if e or f of upward, downward or sideward: here we find a symmetric matrix with 6 different cases (see table 2):

If *e* goes up, than B*e* = C thus for every f, B*ef* = C*f*. So any organisation that would be found through B*ef* will be found through C*f*. So if either *e* or *f* goes upward, we do not need to test B*ef*. This clarifies cases 1, 2 and 3. We will immediately state that we are going to test B*ef* in case that both *e*, *f* go downward. And this clarifies case 6.

cases		*e* goes		
		Upward	Sideward	Downward
***f* goes**	Upward	✓	✓	✓
	Sideward	✓	✓	✓
	Downward	✓	✓	to test

Table 3: If we consider two molecules at the same time and add them to an organisation, this can produce novelty that would not be found through other ways only in one case.

We need now to explore case 4 and 5. Let us suppose that *e* goes sideward, while *f* goes downward. So we are in case 5; do we need to test B*ef*? Since B*e* goes sideward, there exist an organisation C such that B*e* = C with *e* ∉ C and B < C. So B*ef* will be equivalent to test C*ef*. And since C > B than we will simply test this as part of testing C.

Finally let us consider the case where both *e* and *f* go sideward. In this case B*e* = C; B*f* = D. Thus there exist an organisation H such that C ⊔ D = H. Now B*ef* = H*ef*, but B < H, so we do not need to test it as part of B. And the result is that we only need to test sets of molecules such that all the molecules in this set go downward (table 3).

To summarise: So far the algorithm proceeds as follows:

```
Let $L_0$ be the starting sub lattice of organisations;
let O be the set of organisations known;
let Up, Down, Side  be empty dictionaries;
for each organisation B in O:
    for each $m$ not in B ∪ Down[B] ∪ Up[B] ∪  Side[B]:
        calculate B$m$;
        if B$m$ = B:
            Down[B].add($m$); continue;
        if B$m$ not in O:
            calculate $L_{i+1}$ from $L_i$, $B_m$, using the theorems above;
            expand O with $L_{i+1} \setminus L_i$;
        if $m$ not in B$m$:
            for each organisation in $L_{i+1}$, C between B and B$m$:
                Up[C].add($m$)
            Down[B$m$].add($m$)
            Side[B].add($m$)
        else: Up[B].add($m$)
```

At this point a sub lattice L_n will be discovered with a general structure with all the organisations that can be reached by adding one molecule at a time, and applying the union and intersection operations. This is not the complete lattice yet. For each organisation O we need to add the organisations generated by adding to O to every possible set S made up with molecules from Down[O]. In this case we can distinguish in four possible results (one more than before):

1) downward case: O*S* = B;
2) upward case: S ⊂ O*S;*
3) sideward case: S ⊄ O*S,* S ∩ O*S* = ∅;
4) diagonal case: S ⊄ O*S*, S ∩ O*S* ≠ ∅.

The fourth case, the diagonal case, is the new one and happens when S is *partially* contained in O*S*. Again, as we build the sets S from the smallest to the biggest we do not need to check the organisation generated by any set that has a subset T such that O < O*T*. The proof follows the exact same structure as the proof above.

So given any organisation discovered we need to test it by trying to expand it with the molecules not in it, one by one. And with every subset of the “downward” molecules. While we do not need to test a set of “downward” molecules if it contains any subset which is upward, sideward, or diagonal. So the obvious thing to do, is to start with the smaller organisations, with the smaller subsets, and slowly build our way up.

The “No Organisation Left Behind” theorem.

We need now to prove that once the algorithm is followed, every organisation in the lattice of organisations will be found. This can easily be shown.

Suppose by contradiction that C is an organisation in the Lattice that has not been found with the algorithm. Suppose that B is the maximal organisation contained in C found by the algorithm. Such organisation is unique because if there were multiple maximal organisations, the union of them would also be contained in C and would be known because (as part of the algorithm) we are studying the union of all the known organisations. Similarly we assume that C is the minimal organisation unknown above B. So B < C, A maximal organisation known under C, C minimal organisation unknown above B. Now we need to show that following the algorithm we would discover C, against the hypothesis.

Let S be the set of molecules in C, but not in B; S = C\B. Thus obviously B*S* = C.

If we consider any subset T of S, B < B*T* < C. Since B is self-maintaining and C is closed, $B \leq G_O(BT) \leq C$.

Now, as part of the algorithm we explore one by one the subsets of S starting with the smaller one. For each subset T:

either T goes upward letting C be discovered;

or T goes sidewards or diagonal. But in this case there would exist an organisation D = B*T*, with B < D < C. Now if D is known, it would be against the hypothesis that B is Maximal, and if it is unknown, it would be against the hypothesis that C is minimal. So there cannot be a D such that D = B*T*.

Thus for all T, T must go downward, eventually letting us test S and discovering C. Against the hypothesis. So this algorithm explores the full lattice of organisations.

A preliminary version of the code is available at (https://github.com/pietrosperoni/LatticeOfChemicalOrganisations)

Testing the Algorithm on the NTop

To test out the system it was used the NTop Artificial Chemistry (Banzhaf, 1993, 1994). This artificial Chemistry uses boolean vectors of size 4 as molecules, which are then folded into 2x2 matrixes, to react. This results are 16 reacting molecules. One of those acts as an algebraic zero (**0 * a = a * 0 = 0**, for every a) and it is usually eliminated. With the 15 remaining molecules it is possible to obtain 54 organisations out of a space of 32.768 possible subsets. The Brute Force algorithm tests all those subsets. Instead the algorithm developed above was applied. The first step is to take two trivial organisations and the top, o1, and the bottom, o2, were taken. Then the bottom one was expanded, by adding one by one the 15 molecules. The first molecules led to o3. But the 3 organisations (o1, o2, and o3) formed a sub-lattice so it was not possible to expand this further. Second, and third molecules also generated o3. The fourth molecule, added to o2, generated o4. But now it was possible to find o5 as o5 = o3 ⊔ o4. Again o1-o5 formed a sub-lattice. The next molecule generated o6 and permitted to find o7, o8 and o9. o10 lead to o11-o14; o15 all the way to o23. So each organisation found would usually bring others with it, easily calculated. Once the o1-o23 sub-lattice of organisations was found, all the organisations, that could be found by adding a single molecule to o2 had been discovered. Also each molecule tested was divided into downward, upward and sideward, thus simplifying the tests to do later on. Then the algorithm started expanding on those organisations. Expanding o3 did not discover any new organisation. As did o4, o5, …, o9. expanding o10 lead to o24 (and nothing else). Expanding o15 lead to o27, then to o32 and finally to o42. And then nothing else. At this point a sub-lattice was found where adding a single molecule to each of the known 42 existing organisation would always lead back to a known organisation. o1-o42 was not just any sub-lattice, but a sub lattice that could not be pierced by adding a single molecule at a time. Then the algorithm started adding 2 molecules at a time. Expanding o6 it was possible to find o43 which then interacting with the other organisations generated o44 to o54.

It was important to follow the algorithm, not just to understand it better, but because it showed a number of informations about the lattice. First and foremost the fact that the lattice has indeed a number of sub-lattices. Each new organisation found permit us to expand the space of the known organisations to the next sub-lattice in a chain that leads to the complete lattice. Although only 10 basic organisations calculated were necessary to generate the whole lattice, nothing tells us how to find those generators. Indeed finding a minimum number of generators, or just even any set of generators of organisations, is an open problem.

It was also interesting to count how many relations among organisations needed to be calculated, and how many could be derived from the theorems. The results (figure 2) suggests that as the number of organisations grows the number of organisations that needs to be calculated drops following a

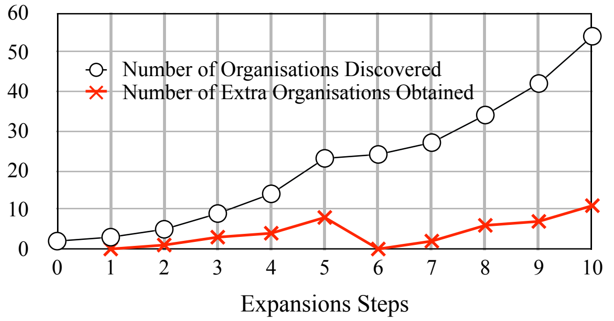


Figure 2: NTop Original: Number of organisations in each subsequent sub-lattice; the complete lattice, 54 organisations, was found in 10 subsequent expansions.

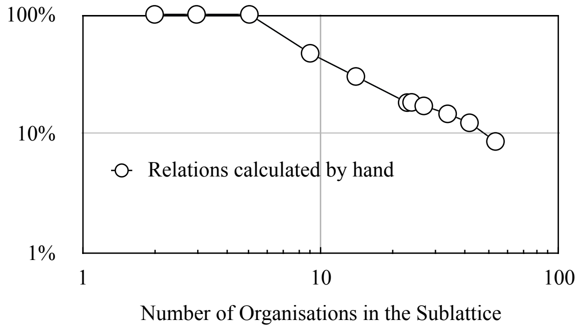


Figure 3: NTop Original: Percentage of Union or Intersection calculated by hand as opposed to extrapolated with the theorems.

power law (making a straight line on a log-log plot). While the remaining relations can all be derived theoretically.

Testing the Algorithm on the Expanded NTop

To try the algorithm on a more challenging artificial chemistry, it was applied to an expansion of the NTop. This time molecules of size 9 were used, which then were folded as 3x3 molecules. This gave 512 molecules. The folding can be done in 9! ways, and this both for the molecule on the right side and on the left side, thus producing 9! * 9! possible Artificial Chemistries. Then the resulting matrix will contain numbers between 0 and 3, and this will be mapped onto the set {0, 1}. This mapping can be done in 2^4 possible ways. The resulting boolean 3x3 matrix must then be unfolded, and this also can be done in 9! ways. So in total there are $2^{4} * (9!)^3 = 1.9 * 10^{17}$ possible Artificial Chemistries. Many of those chemistries produce only a trivial lattice of organisation. For example a lattice that would only contain few organisations, or where every set was an organisation.
In our case, if we consider the molecule
$a = [a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9]$,
the molecules was folded as

$[a_1, a_4, a_7]$
$[a_2, a_5, a_8]$
$[a_3, a_6, a_9]$

when it would react on the left side, and as

$[a_1, a_2, a_3]$
$[a_4, a_5, a_6]$
$[a_7, a_8, a_9]$

when it would react on the right side of the reaction. The result is then transformed according to the map:
$f(x) = \{0 \rightarrow 0;\ 1 \rightarrow 1;\ 2 \rightarrow 1;\ 3 \rightarrow 0\}$
and the resulting boolean matrix was unfolded so that from the matrix:

$[a_{1,1}, a_{1,2}, a_{1,3}]$
$[a_{2,1}, a_{2,2}, a_{2,3}]$
$[a_{3,1}, a_{3,2}, a_{3,3}]$

the vector $[a_{1,1}, a_{1,2}, a_{1,3}, a_{2,1}, a_{2,2}, a_{2,3}, a_{3,1}, a_{3,2}, a_{3,3}]$ was produced. As with the NTop, the algebraic zero, **0** = [0,0,0,0,0,0,0,0,0] was excluded. And in this case also the molecule **1** = [1,1,1,1,1,1,1,1,1] was excluded. The result is an AC with 510 possible molecules, 2^{510} possible sets of molecules; more than 10^{153} sets to test. Obviously too many to directly test them all.

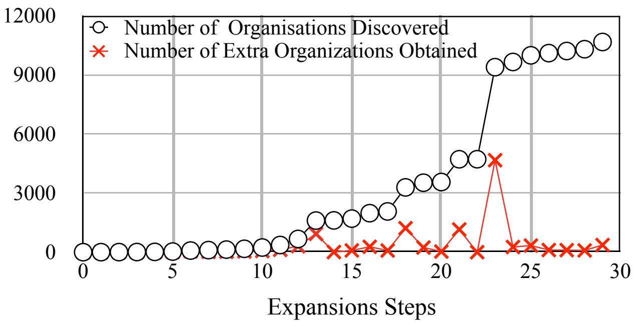


Figure 4: Number of organisations in each subsequent sub lattice.

Results

The algorithm described in this paper was applied. For now it was not possible to find all organisations.
As with the NTop the algorithm started with the lattice which just included the empty and the complete organisations. Then it expanded the set of organisations going through 29 sub-lattices of respectively of 3, 5, 9, 17, 33, 84, 107, 133, 173, 238, 365, 672, 1604, 1612, 1703, 1978, 2066, 3284, 3522, 3557, 4711, 4713, 9377, 9641, 10090, 10196, and 10288 organisations (figure 4). After which no new organisation was found expanding the empty organisation. So 10288 organisations were found just by adding 29 times a single molecule to the empty organisation. This created also two symmetric tables with all the intersection and unions. As the algorithm went on those tables were completed more and more using only the theorems. Again as the number of known organisations grew the percentage of organisations that needed to be calculated by hand decreased following a power law (figure 5). As such the more the lattice is known, the more powerful those theorems are to find the remaining organisations.
Of the 510 molecules 29 permit us to find all the organisations that could be generated by the empty set. Then those organisations started to be expanded themselves.
After the 10288 organisations sub lattice was found the algorithm tested one by one each of those organisations, and for the first 83 organisations, adding a single molecule would keep generating well known organisations. Then on the 84th organisation a molecule was added that expanded the sub lattice, from 10288 to over 69000. And then at 69779 the system could not handle the data using more than 150 gigabyte of RAM and crashed. It should be noted that in a space of 2^{510} molecules it was realistically impossible to map the whole space of all the organisations.

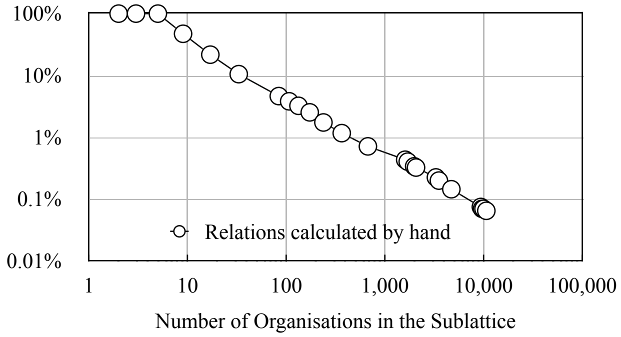


Figure 5: Percentage of Union or Intersection calculated by hand as opposed to extrapolated with the theorems.

Testing the Algorithm against the Brute Force

As a final test the Brute Force Algorithm was applied to this Artificial Chemistry. On a home laptop it only found 263 Organisations before crashing. And the average time to find each organisation was 42 seconds. While the home laptop could find 3000 organisations keeping an average of 0.2 seconds per organisation. In figure 6 the two averages are compared for the first 263 organisations. It should nevertheless be noted that both averages were growing

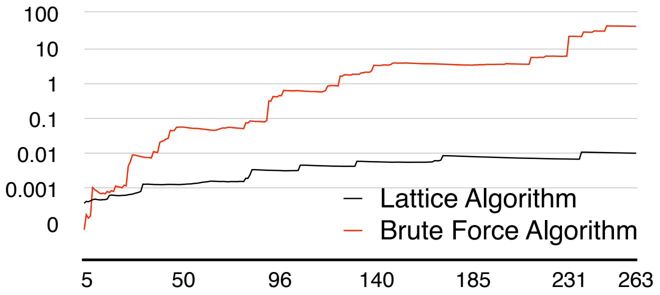


Figure 6: Average time needed to find an organisation, for the first 263 organisations as the number of organisations grows.

although the difference kept increasing. Also it is important to remember that the Brute Force only returns a set of organisations. No informations about the relative relations between the organisations, is returned. What organisations are above or below which others; what is the union or intersections of two organisations, is an information which is equally missing. All information that the Lattice Algorithm can easily return as it needs them to compute the lattice. So not only is the Brute Force much slower and generally un-efficient algorithm. But it also is insufficient to really let us know an artificial chemistry.

Consequences and Conclusions

Reaction networks appear everywhere. And in their exploration the study of organisations, and their lattice is a necessary step to really understand their global behaviour as constructive dynamical system. The fact that organisations form a Lattice permit us to compute them faster. Although several results claim to be able to find all the organisations of a reaction network (Centler at al 2008, Centler at al 2010) they never used the algebraic properties of the Artificial Chemistry (namely that it is a lattice).

Another important aspects that was uncovered in this work is the concept of sub-lattice, as a subset of organisations such that each union and intersection is an organisation inside the sub-lattice. From mathematics we know that if we consider the space of all sub-lattices of a lattice, they form a lattice, too: the lattice of sub-lattices of the lattice of organisations. Such lattice is not explored here, instead we merely extract a chain inside such lattice of sub-lattices and use this to build the lattice of organisations. Interestingly on the NTop system we could identify a sub lattice of organisations that was closed respect to the operation of adding a single molecule to any organisation. This shows that if we must find all the organisations, we cannot simply consider adding 1 molecule.

But it also means that in some situations we are not interested in the complete lattice of organisations, but only in a sub-lattice since the the artificial chemistry will only explore that. For example in a Flow System both organisations and semi organisations form a lattice. But while each organisation is a semi-organisation the opposite is generally not true. Thus the lattice of organisations is a sub-lattice of the lattice of semi-organisation. And this fact could be used to map it.

Similarly, Artificial Chemistries are not the only case of lattice present in the fields of Bioinformatics, Artificial Life and Systems Biology. Researchers looking at autocatalytic cycles and closed sets also are looking at sets of molecules that form lattices. Thus the same procedures that were exposed here, and the same theorems, can be applied over there, with comparable results.

Every research clarifies some elements, while opening new questions.

- In particular it is still unclear what is the most effective way to apply the lattice theorems to study the lattice of organisations. Yes, theorems can shortcut the calculations, and we could see that the algorithm was at least three orders of magnitude faster than the Brute Force algorithm (and the difference was increasing), but finding which theorems can be applied can be time consuming as well. So a smart strategy might need to be applied to chose when to try to apply the theorems, as a further improvement of the algorithm.
- Also it is unclear why the number of times the theorems are not applicable follows so closely a power law. This might be related to the nature of the Lattice of Organisations as a graph. But the details are still missing.
- The roles of sub-lattices, what is the sub-lattice of sub-lattices, and how the sub-lattices of the lattice or organisations can be used to study an ecology of different experiments on one artificial chemistry is also an open question.
- And finally the artificial chemistry presented here is very vast and exploring it all is at the moment impossible. It could as such be used as a benchmark for future work.

Acknowledgement: The author acknowledges support from the European Union through funding under FP7–ICT–2011–8 project HIERATIC (316705). Also thanks Peter Dittrich for personal remarks and the two anonymous reviewer for fundamental comments.

References

Banzhaf W. (1993) Self-replicating sequences of binary numbers. Foundations II: Strings of length N- 4 *Biol. Cybern*, 69, pg 275-281

Banzhaf W. (1994) Self-replicating sequences of binary numbers. Foundations I: The Built Up of Complexity. *Complex Systems*, 8, pg 215-225

Centler, F, Kaleta C., Speroni di Fenizio, P; Dittrich, P (2008) Computing chemical organizations in biological networks. *Bioinformatics* 24 (14): 1611-1618.

Centler, F.; Kaleta C.; Speroni di Fenizio, P.; Dittrich, P. (2010) A parallel algorithm to compute chemical organizations in biological networks. *Bioinformatics Applications Note* 26 (14), pg 1788 - 1789

Dittrich, P., Speroni di Fenizio, P (2007). Chemical organization theory. *Bull. Math. Biol.*, 69(4): pg 1199-1231

Fontana, W. and Buss, L. W. (1994). 'The arrival of the fittest': toward a theory of biological organization. *Bull. Math. Biol.*, 56: pg 1-64

Fontana, W. and Buss, L. W. (1994). What would be conserved if 'the tape were played twice'?. *Proc. Natl. Acad. Sci. USA*, 91: pg 757–761

Speroni di Fenizio, P.; (2007). *chemical organizations theory.* Doctoral thesis, Chair of Systems Analysis, Department of Computer Science, University of Jena, Germany, 2007

Fontana, W. and Buss, L. W. (1996). The barrier of objects: from dynamical systems to bounded organizations. Casti J. and Karlqvist A., editors, *Boundaries and Barriers*, pg 56–116, Addison-Wesley

Coevolution of cooperation and layer selection strategies in multiplex networks

Katsuki Hayashi[1,2] Reiji Suzuki[1,3] and Takaya Arita[1,4]

[1]Graduate School of Information Science, Nagoya University
[2]lin@alife.cs.is.nagoya-u.ac.jp,[3]reiji@nagoya-u.jp,[4]arita@nagoya-u.jp

Abstract

Recently, the emergent dynamics in multiplex networks, composed of layers of multiple networks, has been discussed extensively in network sciences. However, little is still known about whether and how the evolution of strategy for selecting which layer to participate in can contribute to the emergence of cooperative behaviors in multiplex network of social interactions. For this purpose, we constructed a coevolutionary model of cooperation and layer selection strategies in which each individual selects one layer from multiple layers of social networks and play Prisoner's Dilemma with neighbors in the selected layer. We found that proportion of cooperative strategies increased with increasing the number of layers regardless of the degree of dilemma, and this increase occurred due to the cyclic coevolution processes of game strategies and layer selection strategies.

Introduction

The recent progress in network sciences revealed that structures of interactions among individuals can affect the emergence and evolution of cooperative behaviors significantly (Nowak and May (1992), Ohtsuki et al. (2006)). This is because local interactions allow cooperative clusters to grow in the population of defectors in general (Nowak and May (1992)). While most of previous studies assumed that all individuals interact in a network of a single social relationship or context, there exist a bunch of networks of social interactions in a real world, and it is expected that they are affecting with each other directly or indirectly in various ways.

Such a situation of interactions among networks is known as a kind of multiplex networks, multilayer networks, interdependent networks or networks of networks, which have been discussed extensively in network sciences (Kivelä et al. (2014)), recently. A pioneering study showed that properties of cascading failures on interdependent networks differ significantly from those of single-network systems, in that the existence of inter-connecting links between networks changes the threshold and the order of transition for cascading failures (Buldyrev et al. (2010)).

There are various situations in which cooperative behaviors can evolve in multiplex networks of social interactions. Wang et al. discussed interactions between the evolution processes of cooperative behaviors in two interdependent networks (Wang et al. (2013)). In addition to the total payoff obtained from the Prisoner's Dilemma game (PDG) with its neighbors in a two-dimensional regular network, each individual obtains the additional payoff, that is the payoff obtained by another individual at the corresponding position in the other network, reflecting indirect and interdependent effects of a network on the other. They showed that the intermediate degree of interdependence contributed to the evolution of cooperation. They also showed that the degree of interdependence can self-organize to the optimal value (Wang et al. (2014)) through the individual-level adaptation of it.

In this study, we focus on the fact that an individual can belong to multiple kinds of social networks, which is a ubiquitous situation in a real life situation. We can expect that the behavior of an individual in one network can affect its future behavior in another network. Gomez-Gardenes et al. assumed that each individual belongs to multiple random networks and has a strategy of PDG (cooperate or defect) for each network that is called a layer. The population evolves according to the fitness determined by the accumulated payoff of the games with neighbors in all the layers (Gómez-Gardenes et al. (2012)). They found that the evolution of cooperation was facilitated by the multiplex structure only when the temptation to defect was large. However, it seems not natural to assume that all individuals always participate in games in all the networks, because there exist physical, social and temporal constraints in a real life situation. Instead, it is more natural to assume that each individual actively select which network to participate in depending on the state of interactions, and such a layer selection strategy can coevolve with game strategies.

Our purpose is to clarify whether and how the evolution of layer selection strategy can contribute to the emergence of cooperation in a multiplex network of social interactions. We assume multiple layers composed of random networks. Each individual belongs to all the layers but selects one layer and play games with neighbors in the selected layer. Both the layer selection strategy and the strategy for PDG for each layer coevolve according to the fitness based on the payoff from the games. We show that the larger the number of layers, the larger the proportion of cooperators increases, implying that multiplex networks can contribute to the evolution of cooperative behaviors. We also show it was caused by the dynamic coevolution process of strategies through which a burst of the proportion of individuals occurred in different layers repeatedly.

Model

Multiplex Network

Fig. 1 shows a schematic image of the model and Algorithm 1 also shows a pseudo-code of our model. There are M layers that abstract different channels or contexts of social interactions among individuals. For example, each layer corresponds to a social relationship in a real group or a friendship in a social networking service (SNS) in the Internet. Each layer is composed of a network of interactions among individuals in the corresponding relationship. An individual i is represented as a node n_i^l $(i = 0, 1, ..., N-1)$ in the layer l $(l = 0, 1, ..., M-1)$, and thus it is represented as a set of nodes $\{n_i^0, \cdots, n_i^{M-1}\}$. The existence of a link between the individual i and j in the layer l means that i and j are neighboring individuals who can interact with each other in the layer l. In this study, the topology of each layer is defined as an Erdös-Rényi (ER) random graph with the average degree k. It is known that cooperative behavior is not easy to evolve in ER random graphs. We adopt this structure in order to see if increasing the number of layers can contribute to evolution of cooperation in spite of such a hard situation. Each time step is composed of two phases: playing games and updating strategies, as explained below. Hereafter, we describe that an individual i is in the layer l if it selects the layer l $(sl_i = l)$, in that it participates in interactions in the social network represented by the layer l.

Playing games

We assume that each individual can participate in interactions in only one layer at every time step, reflecting the physical, temporal and cognitive constraints. Thus, each individual i has a layer selection strategy $sl_i \in \{0, 1, \cdots, M-1\}$. It determines the layer in which the individual i plays PDG with its neighbors. Hereafter, we describe that an individual i is in the layer l if it selects the layer l $(sl_i = l)$, in that it participates in interactions in the social network represented by the layer l. Each individual i also has a strategy for PDG sp_i^l (cooperate (C) or defect (D)) for each layer l. It plays a PDG using the strategy $sp_i^{sl_i}$ with each neighboring individual j, in its selected layer sl_i, who is in the same layer $(sl_j = sl_i)$ and plays $sp_j^{sl_j}$. The payoff matrix of the PDG is defined in Table 1. In addition, if there is a neighbor (j), in its selected layer sl_i, who is in a different layer $(sl_j \neq sl_i)$, the individual i gets an additional payoff d regardless of the game strategy of that neighbor j. d represents the payoff that can be earned by the individual alone instead of playing a game. We defined the parameter d in order to assume that not playing game is better than mutual defection but worse than mutual cooperation. This condition is satisfied if $0 < d < 1$. The total payoff P_i is regarded as the fitness of the individual i. For example, in Fig. 1, the individual i chooses the layer 2 and there are 2 neighbors in its selected layer. It obtains the payoff 1 by cooperating with a cooperator in the same layer, and earns a payoff d alone. As a result, it obtains the fitness $1 + d$.

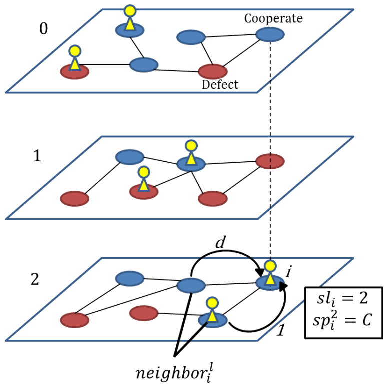


Figure 1: A schematic image of our model.

Table 1: A payoff matrix of PDG. b represents the temptation to defect.

sp_j sp_i	cooperate (C)	defect (D)
cooperate(C)	1	0
defect(D)	b	0

Updating strategies

Each individual i updates its PDG strategy in the selected layer $sp_i^{sl_i}$ and its layer selection strategy sl_i according to the fitness after playing games. We assume that individuals can obtain the information about the fitness and PDG strategies of neighboring individuals in the selected layer before updating strategies. The value of PDG strategy sp_i^l in the next time step nsp_i^l is determined by the following procedure.

i One individual j is randomly selected from its neighboring individuals in the layer sl_i $(neighbor_i^{sl_i})$ regardless of sl_j.

ii If the fitness of the individual j (P_j) is higher than its own fitness P_i, $nsp_i^{sl_i}$ will be $sp_j^{sl_i}$ $(nsp_i^{sl_i} \leftarrow sp_j^{sl_i})$ with the following probability used in (Gómez-Gardenes et al.

(2012)):

$$\prod_{i\leftarrow j} = \begin{cases} \frac{P_j - P_i}{b\max(|neighbor_i^{sl_i}|, |neighbor_j^{sl_j}|)} & \text{if } P_j > P_i \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

This equation means that each individual imitates the strategy of j with a positive probability if the fitness of the neighbor j is higher than its own fitness. The actual value of imitation probability depends on the difference between these fitness: the individual i always imitate the strategy of j when its fitness is the minimum and the fitness of j is the maximum. This imitation probability linearly decreases as the difference between these fitness values decreases. Otherwise, if the fitness of the individuals i is smaller than that of the neighbor j, it does not change the strategy ($nsp_i^{sl_i} \leftarrow sp_i^{sl_i}$). This means that it can imitate a PDG strategy of a better neighbor in its selected layer.

iii nsp_i^l is replaced with C or D randomly with a mutation probability μ.

The layer selection strategy sl_i in the next time step nsl_i is determined by the following procedures.

i One individual j is randomly selected from its neighboring individuals in its selected layer sl_i ($neighbor_i^{sl_i}$) regardless of sl_j.

ii If the fitness of the individual j (P_j) is higher than its own fitness P_i, nsl_i will be sl_j ($sl_i \leftarrow sl_j$). Otherwise, it does not change the strategy ($nsl_i \leftarrow sl_i$). This means that it can imitate a layer selection strategy of a better neighbor in its selected layer, which allows an individual to move to a different layer.

iii nsl_i is replaced with a random value from $\{0, 1, ..., M-1\}$ with the mutation probability μ.

Finally, all the strategies are updated simultaneously ($sp_i^{sl_i} \leftarrow nsp_i^{sl_i}$ and $sl_i \leftarrow nsl_i$, for all i).

In some situations, it might be plausible to assume that changing a group or network to which an individual belongs (i.e., its layer selection strategy) is easier than changing the strategy related to its personality (i.e., its game strategy). The processes described above reflects such a situation in which changes in the layer selection strategies can happen more frequently than changes in the game strategies.

Algorithm 1 A pseudo-code of our model. $payoff(a, b)$ represents the payoff value obtained by an individual who plays a strategy a with an opponent playing a strategy b. $neighbor_i^l$ represents the set of the neighboring individuals of the individual i in the layer j. $rnd(s)$ represents a function that returns a randomly selected element from the set s. rnddist() also represents a function that returns a random value from the uniform distribution [0, 1].

```
(initialize strategies)
for $i = 0 \rightarrow N-1$ do
    for $j = 0 \rightarrow M-1$ do
        $sp_i^j \leftarrow rnd(\{C, D\})$
    end for
    $sl_i \leftarrow rnd(\{0, 1...M-1\})$
end for
for $t = 0 \rightarrow G-1$ do
    for $i = 0 \rightarrow N-1$ do
        (playing games)
        $P_i \leftarrow 0$
        for each $neighbor_i^{sl_i}$ do
            if $sl_i == sl_j$ then
                $P_i \leftarrow P_i + payoff(sp_i^{sl_i}, sp_j^{sl_j})$
            else
                $P_i \leftarrow P_i + d$
            end if
        end for
    end for
    (updating strategies)
    for $i = 0 \rightarrow N-1$ do
        (updating a PDG strategy)
        $j \leftarrow rnd(neighbor_i^{sl_i})$
        if $P_i < P_j$ then
            if rnddist() < $|\frac{P_j - P_i}{b\max(|neighbor_i^{sl_i}|, |neighbor_j^{sl_j}|)}|$ then
                $nsp_i^{sl_i} \leftarrow sp_j^{sl_j}$
            else
                $nsp_i^{sl_i} \leftarrow sp_i^{sl_j}$
            end if
        end if
        (updating a layer selection strategy)
        $j \leftarrow rnd(neighbor_i^{sl_i})$
        if $P_i < P_j$ then
            $nsl_i \leftarrow sl_j$
        else
            $nsl_i \leftarrow sl_i$
        end if
    end for
```

Algorithm 1 continued

```
        (mutation)
        for $i = 0 \to N-1$ do
            if rnddist() $< \mu$ then
                $nsp_i^{sl_i} \leftarrow rnd(\{C, D\})$
            end if
            if rnddist() $< \mu$ then
                $nsl_i \leftarrow rnd(\{0, 1...M-1\})$
            end if
        end for
        for $i = 0 \to N-1$ do
            $sp_i^{sl_i} \leftarrow nsp_i^{sl_i}$
            $sl_i \leftarrow nsl_i$
        end for
    end for
```

Experiments and Discussion

We conduced experiments on the constructed model for the purpose of revealing the co-evolution dynamics between the layer selection strategy and the cooperative behavior in multiplex networks. We used the following values as the experimental parameters: $N = 100$, $M = \{1, 3, ..., 19\}$, $k = 10.0$, $b = \{1.1, ..., 2.1\}$, $G = 10000$, $\mu = 0.02$ and $d = 0.4$. sp_i^l and sl_i were initialized with random values from their domains in the initial population. We used $d = 0.4$, which is an intermediate value in its domain ($0 < d < 1$), in order to see the basic effect of this parameter on the evolution process. The experiment results are the average of 5 trials for each combination of the parameter settings of M and b.

We aim to understand how the proportion of cooperative behaviors can be changed by the increase in the number of layers M. First, we define the selected strategy of an individual i as the one used by it in its selected layer (i.e., $sp_i^{sl_i}$), and focus on quantitative effects of M on the proportion of cooperation among the selected strategies of all the individuals R_c. We plot the average of R_c over all generations with different combinations of M and b, as a heat chart, in Fig. 2. The horizontal axis shows the number of layers M and the vertical axis shows the temptation to defect b. We see that R_c increased with increasing M and decreasing b. This means that the multiplex network facilitated the evolution of cooperation in any conditions of the Prisoner's Dilemma.

More specifically, R_c greatly decreased with increasing b when M was small. However, when M was large enough, R_c kept relatively large (or only slightly decreased) with increasing b. Thus, we can say that the negative effect of b on cooperative strategies could be reduced significantly by increasing the number of layers M.

Next, we plot the entropy of the probability distribution of sl_i, as a measure for the degree of dispersion of individuals over the networks in Fig. 3. We see that the entropy increased drastically with increasing M. This means that the wider distribution of individuals over more layers might contribute to the evolution of cooperation. We also see that the entropy slightly decreased with decreasing b, which implies that higher but not too high values of the entropy contributed to the evolution of cooperation.

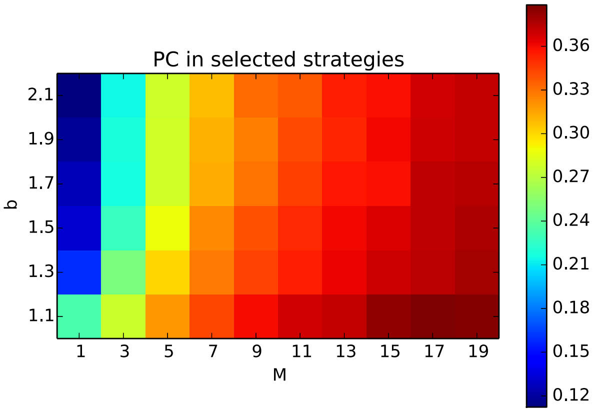


Figure 2: The heat chart of proportion of cooperative behaviors among the selected strategies $sp_i^{sl_i}$ (R_c).

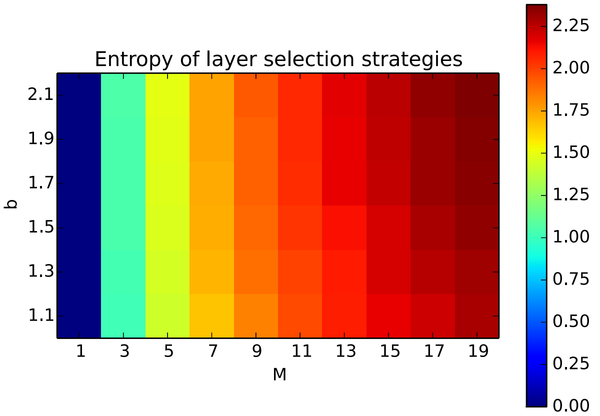


Figure 3: The entropy of the probability distribution of layer selection strategies sl_i.

Then, we focus on the transitions of the proportion of individuals in each layer and the proportion of cooperation in the selected strategies among them. We plot the transition of these indices from 2000th to 3000th step in typical trials when $b = 1.7$ and $M = 1$ (Fig. 4), 3 (Fig. 5) and 9 (Fig. 6). We focus on this period in order to observe the typical transitions after the transient process from the initial population. There are M panels, each corresponding to a layer. The horizontal axis represents step, the blue line represents the proportion of cooperation among selected strategies ($sp_i^{sl_i}$) of individuals in the corresponding layer. The red line represents the proportion of individuals that selected the corre-

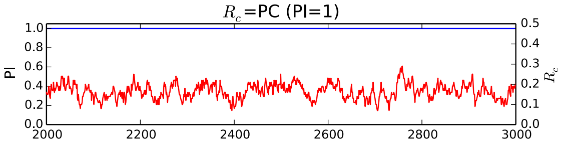


Figure 4: The transition of the proportion of cooperative behaviors among the selected strategies $sp_i^{sl_i}$ (R_c) when $M = 1$.

sponding layer. In addition, there is an additional panel in the bottom, which shows the average proportion of cooperation in the selected strategies.

When $M = 1$ (Fig. 4), all individual exist in a single layer ($PI = 1$). The proportion of cooperators slightly fluctuated at small values around 0.15. This can be said to be the baseline behavior of a standard model for the evolution of cooperation in a single and random network.

On the other hand, when $M = 3$ (Fig. 5), the average proportion of cooperators was higher than that when $M = 1$, which fluctuated at around 0.25. We also see that the proportion of individuals in each layer largely fluctuated and often reached very high values. This means that the individuals were distributed all over the layers, but they often got together in a layer.

Furthermore, when $M = 9$ (Fig. 6), the average proportion of cooperators became around 0.3, which was higher than that when $M = 3$. The occurrence of a burst-like rise and fall of the number of individuals in a layer was more pronounced and it often reached its peak around 0.8, meaning that most of individuals have selected the same layer. On the other hand, the proportion of individuals in the other layers tended to be much smaller than 0.2. We also see the gradual increase and the rapid decrease in the proportion of cooperators before and after the burst of the proportion of individuals, respectively.

The reason for this evolutionary dynamics that facilitated the cooperation can be summarized as follows. In this model, there are no games between individuals in different layers. Thus, the smaller the proportion of individuals in a layer is the higher the locality of interactions is, because it decreases the number of links used for playing games in effect. It has been pointed out that the higher locality for the smaller number of links can facilitate the evolution of cooperation (Ohtsuki et al. (2006)). Thus, cooperators can invade into a layer with the smaller number of individuals gradually. Such cooperative relationships in the layer make individuals in other less-cooperative layers (after a burst of the number of individuals) select the focal layer, which brings about a rapid increase in the proportion of individuals in the layer. However, this allows defectors to invade into the focal layer, and thus the proportion of cooperators decreases rapidly. In such a population of defectors, individuals select other cooperative layers because it is better not to play game with neighbors than to play games with many defectors. This further causes another burst of the proportion of individuals in another cooperative layer.

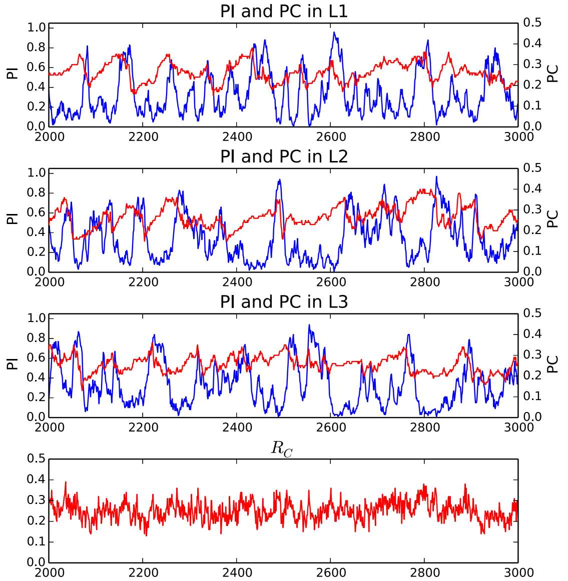


Figure 5: The transition of the proportion of cooperative behaviors (PC) and the proportion of individuals (PI) in each layer and proportion of cooperative behaviors among the selected strategies $sp_i^{sl_i}$ (R_c) when $M = 3$.

In Fig. 7 we plot the trajectory of these two indices in the 1st layer in Fig. 6 from 2000th to 4000th step. The horizontal axis represents the proportion of individuals in the layer and the vertical axis represents the proportion of cooperative strategies among the selected strategies in the layer. We see that the cyclic coevolution process of these indices occurred repeatedly, as explained above.

In addition, we conduced further experiments with different values of the parameter d. The general trend of evolution process did not change but the burst of the number of individuals occured more frequently with increasing d. This is expected to be due to the fact that individuals are easier to move to another layer by avoiding mutual defections as d increases.

Overall, repeated occurrences of this dynamic coevolution process of game strategies and layer selection strategies are expected to maintain the high proportion of cooperators in the whole population.

Conclusion

We constructed a coevolutionary model of cooperation and layer selection strategies in which each individual selects one layer from multiple layers of social networks and play Prisoner's Dilemma with neighbors in the selected layer, for the purpose to clarify whether and how the evolution of layer

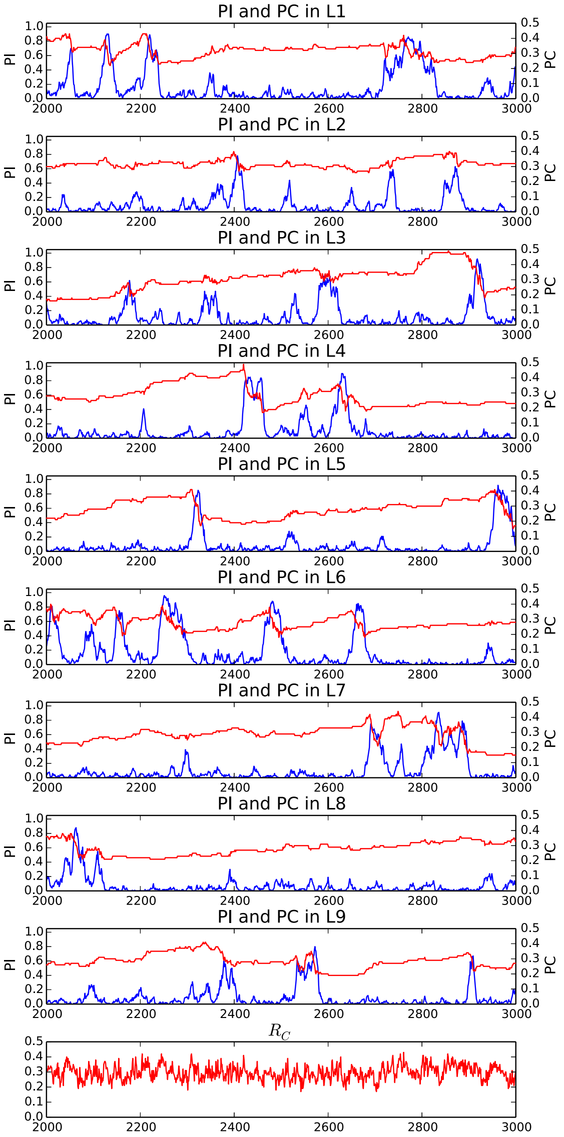


Figure 6: The transition of the proportion of cooperative behaviors (PC) and the proportion of individuals (PI) in each layer and proportion of cooperative behaviors among the selected strategies $sp_i^{sl_i}$ (R_c) when $M = 9$.

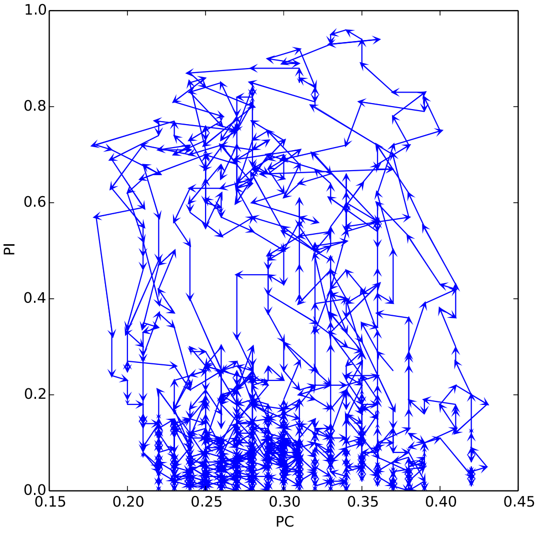


Figure 7: The trajectory of the proportion of cooperative behaviors (PC) and the proportion of individuals (PI) in the 1st layer in Fig. 6.

selection strategy can contribute to the emergence of cooperative behaviors in multiplex network of social interactions. From the results of experiments, we found that the proportion of cooperative strategies increased with increasing the number of layer regardless of the degree of dilemma, and this increase occurred due to the cyclic coevolution processes of game strategies and layer selection strategies. The emergence of such a cyclic process has been pointed out by the study of coevolution between cooperative behavior and network structures interaction in which the network rewiring strategies can coevolve with the game strategies (Suzuki et al. (2008)). Thus, this implies that such a dynamic process could be common phenomenon in a real world. It should be noticed that we further clarified that such a dynamic process can strongly facilitate the evolution of cooperation, in this paper.

Future work includes verifying whether the similar process can occur when the topology of networks is changed (e.g., scale-free or small world).

References

Buldyrev, S. V., Parshani, R., Paul, G., Stanley, H. E., and Havlin, S. (2010). Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028.

Gómez-Gardenes, J., Reinares, I., Arenas, A., and Floría, L. M. (2012). Evolution of cooperation in multiplex networks. *Scientific reports*, 2.

Kivelä, M., Arenas, A., Barthelemy, M., Gleeson, J. P., Moreno, Y., and Porter, M. A. (2014). Multilayer networks. *Journal of Complex Networks*, 2(3):203–271.

Nowak, M. A. and May, R. M. (1992). Evolutionary games and spatial chaos. *Nature*, 359(6398):826–829.

Ohtsuki, H., Hauert, C., Lieberman, E., and Nowak, M. A. (2006). A simple rule for the evolution of cooperation on graphs and social networks. *Nature*, 441(7092):502–505.

Suzuki, R., Kato, M., and Arita, T. (2008). Cyclic coevolution of cooperative behaviors and network structures. *Physical Review E*, 77(2):021911.

Wang, Z., Szolnoki, A., and Perc, M. (2013). Optimal interdependence between networks for the evolution of cooperation. *Scientific reports*, 3.

Wang, Z., Szolnoki, A., and Perc, M. (2014). Self-organization towards optimally interdependent networks by means of coevolution. *New Journal of Physics*, 16(3):033041.

Volatility and spatial distribution of resources determine ant foraging strategies

Drew Levin[1], Joshua P. Hecker[1], Melanie E. Moses[1,2,3], Stephanie Forrest[1,3]

[1] Department of Computer Science, University of New Mexico, Albuquerque, NM, USA
[2] Department of Biology, University of New Mexico, Albuquerque, NM, USA
[3] Santa Fe Institute, Santa Fe, NM, USA
drew@cs.unm.edu

Abstract

Social insect colonies have evolved collective foraging strategies that consist of many autonomous individuals operating without centralized control. The ant colony optimization (ACO) family of algorithms mimics this behavior to approximate solutions to computationally difficult problems. ACO algorithms focus on pheromone recruitment, which is only one of several known biological foraging strategies. Here, we use a spatial agent-based model to simulate three foraging strategies: pheromone recruitment, nest recruitment, and random search. We compare their performance across two environmental dimensions: spatial distribution of food resources and resource volatility. We find that pheromone recruitment performs only marginally better than the simpler nest recruitment strategy in most environments. Further, both strategies become progressively less efficient as resource dispersion and volatility increase. In the extreme, with highly dispersed or volatile resources, the simplest strategy of all, random search, outperforms the other two. Our results suggest that in many environments, pheromone-based strategies may not be required and that simpler methods like random search or nest recruitment may be sufficient, both for biological ants and computational methods.

Introduction

Social insects are notable for their ability to harness large populations of simple individuals without any apparent centralized control to solve complex problems, such as finding food and building nests (Hölldobler and Wilson, 1990). Computer scientists have long been interested in ant foraging behaviors, particularly as inspiration for *ant colony optimization* (ACO) search and optimization algorithms (Dorigo et al., 2006). These algorithms are based on one aspect of ant foraging called *stigmergy* (Theraulaz and Bonabeau, 1999). In this context, stigmergy refers to ants that alter their environment by depositing chemical markers called pheromones to indicate promising search directions. As they accumulate, the chemical markers form a trail leading from the ants' nest to a food source. ACO algorithms have been applied to many computational problems, including network routing (Di Caro and Dorigo, 1998), the Traveling Salesman Problem (Dorigo and Gambardella, 1997), and task scheduling (Merkle et al., 2002).

However, recent work has shown that chemical recruitment strategies in isolation may be suboptimal in the wrong environment. Evison et al. (2008) show that visual landmarks and pheromone contribute equally to an ant's ability to locate a previously encountered food source, and that the two may have a complementary function. Pheromone trails may also lead to suboptimal solutions, directing ants (or algorithms) to lower-quality resources before a richer location can be detected (Beckers et al., 1990; Robinson et al., 2008), although these traps may be avoided using repellent pheromone (Czaczkes, 2014). The drawbacks to chemical recruitment are not limited to ant foraging, nor to the problem of reinforcing the wrong path in a stable environment. Previous work has shown that T cells that rely on chemical gradients to locate sources of infection can become stuck when the infection spreads faster than the signal can diffuse (Levin et al., unpublished). The drawbacks of chemical recruitment strategies suggests that successful use of ACO requires an understanding of the appropriate domains where these algorithms are applied, and which other foraging strategies might be leveraged in different domains. There is relatively little work which investigates other ant foraging techniques or classifies which environments are most appropriate for which foraging strategies (Pratt, 2008; Schmolke, 2009; Pinter-Wollman et al., 2012).

There are many different foraging strategies employed by different ant species (Lanan, 2014). The desert ant, *Aphenogaster cockerelli*, forages individually with site fidelity (Sanders and Gordon, 2002). Site fidelity may be a more effective foraging strategy than pheromone recruitment in some contexts (Letendre and Moses, 2013). The acorn ant, *Temnothorax albipennis*, uses tandem running where informed ants that have located a food source lead naive ants to the food, without using any detectable pheromone trail (Franks and Richardson, 2006). Ants such as *Formica cinerea* establish long-term trunk trails, where massive numbers of ants follow one another to stable sources of food (Markó and Czechowski, 2012). The predatory *Pheidologeton diversus* raid smaller ant species and termites when colonies are discovered (Moffett, 1988).

In this paper, we explore the hypothesis that the best foraging strategy is determined in large part by the environment in which the ants live. We focus on how food resources are distributed and their temporal variability, which we call *volatility*. This hypothesis is supported by the observation that animal species use different strategies in different environments (Kacelnik and Bateson, 1996).

We investigate this hypothesis with an agent-based model of ant foraging. We model three different ant-based foraging strategies, which subsume most strategies observed in nature: solitary random walk, nest recruitment, and pheromone recruitment. We quantify the efficacy of each strategy across two environmental dimensions, the spatial layout of resources and the volatility of resources, and we evaluate the ability of each strategy to adapt to different environments.

We find that both pheromone and nest recruitment perform best in clustered stable environments, and their efficiency declines as food dispersion and volatility increase. To be effective, the recruitment strategies each require that the ants complete at least two round trips to a location before the food disappears (volatility). Because random search has no memory, volatility does not strongly affect its performance, and it performs well in environments where food is highly dispersed.

Recruitment strategies that have been optimized for one environment can be detrimental in other environments. For example, when we optimized the (nest and pheromone) recruitment strategies for stable environments, they performed poorly when volatility was increased, and in the extreme worse than random search. However, in environments with upredictable volatility, nest and pheromone recruitment strategies outperform random search, suggesting that recruitment is a powerful mechanism even in highly randomized environments. Finally, in environments where resource locations are sufficiently predictable, pheromone-based strategies are efficient, similar to Flanagan et al. (2011). In most cases, however, nest recruitment performs similarly to pheromones and requires a simpler mechanism (local interaction).

In addition to specific insights about ant foraging strategies, our results suggest that new approaches could be adapted into ACO algorithms. Since phereomone-based recruitment is nontrivial to implement in a fully distributed artificial system, nest recruitment could be an attractive alternative. Moreover, the modeling approach used here could be used to classify more generally which distributed search strategies perform best in which environments.

Model Description

We developed an agent-based model to study the effectiveness of various foraging strategies in different environments (Fig. 1), focusing on the spatial and temporal distribution of food resources on a flat surface. Our model extends the central-place foraging algorithm (CPFA) and swarm robotics platform studied in Hecker and Moses (2015), by including volatile resources and new foraging strategies.

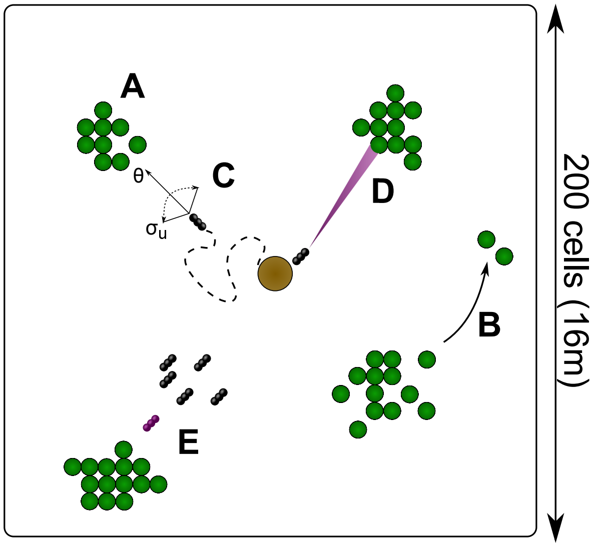


Figure 1: **Model Conceptualization.** The model is initialized on a square grid with the nest in the center. **A)** Food is randomly placed in patches a minimum distance from the nest. **B)** Food from a single patch moves at a constant rate to a new patch (volatility). **C)** Ants perform a random walk from the nest. An ant's angular trajectory θ is varied at each time step by choosing from a normal distribution: $N(\theta, \sigma_u)$. **D)** An ant lays a pheromone trail (decays exponentially) to a specific location if it detects enough food in its vicinity. **E)** Ants returning from a successful foraging search recruit ants resting at the nest.

The model consists of a two-dimensional grid with discrete food units (analogous to seeds) placed on the grid before the run begins. There is a single nest, where ants congregate, leave to search for food, return with food if successful, and possibly recruit other ants to follow them to a food source. Food sources can be arranged in different distributions (e.g., grouped together in a small number of clumps or dispersed randomly across the environment). Volatility is modeled as movement—food moves at a rate parameterized by the model. In the experiments for this paper, we varied the number of clustered food piles and the rate at which piles move, while holding the total amount of food constant (Fig. 1 A and B).

A recent paper by Lanan (2014) catalogs certain ant foraging strategies used by monodomous (single nest) ant species: random search, site fidelity, tandem running, pheromone recruitment, nest recruitment, and trunk trails. Two of these strategies, tandem running and trunk trails, closely resemble other strategies: nest recruitment is similar to tandem running and pheromone recruitment is similar to trunk trails. Therefore, we model three distinct foraging strategies: *random search*, *pheromone recruitment*, and *nest recruitment* (Fig 1 C-E), where each strategy is designed to mimic for-

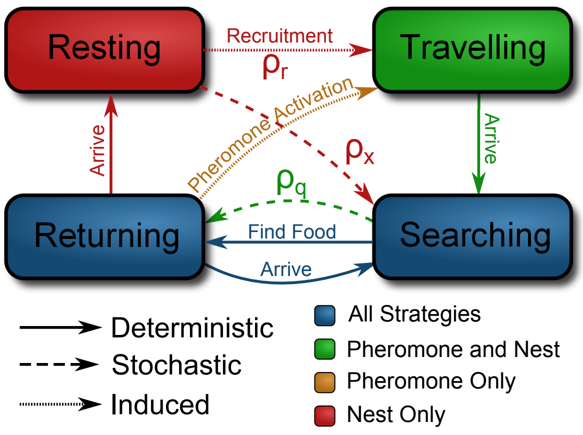


Figure 2: **Ant Foraging States.** Ants are initialized at the nest in either the `resting` (with nest recruitment) or `searching` state. Ants transition between behaviors based upon cues from the environment, random chance, and stimulation by other ants or pheromone.

aging strategies used by real ants (Moffett, 1988; Hölldobler and Wilson, 1990; Sanders and Gordon, 2002; Markó and Czechowski, 2012). We enable site fidelity in both recruitment strategies based on field observations.

- **Random Search:** Individual ants leave the nest all at once and perform a correlated random walk through the two-dimensional space. Ants continue searching until they encounter food. Ants that encounter food pick it up, return to the nest, and begin a new search. The correlated random walk works by choosing a new trajectory for each ant from a normal distribution $N(\theta, \sigma_u)$, centered on the ant's current trajectory θ. Ants using the random search strategy have no memory and perform no recruitment.

- **Pheromone Recruitment** is implemented following the central-place foraging algorithm (CPFA) detailed in Hecker and Moses (2015). Ants leave the nest and search randomly as described above. Ants may give up searching at any time with a probability ρ_q and return to the nest. However, if an ant finds food at any point it picks it up and immediately checks the neighboring 8 grid cells for more. Next, it decides to reinforce the location with probability $P(k; \lambda_p)$, where P represents the cumulative Poisson distribution and k is the amount of food found in the immediate neighborhood. The ant may also use site fidelity to return to the previously visited location with probability $P(k; \lambda_f)$. It then returns to the nest, creating the trail upon its arrival if $P(k; \lambda_p) > U(0, 1)$; the trail decays at a rate of σ_p. Subsequent ants may follow this trail to the same location before the trail evaporates. Recruited ants perform an informed correlated random walk upon arrival and may also lay a pheromone trail back to the nest.

Abbr.	Name	Distribution
σ_u	Uninformed Search Correlation	$U(0, 2\pi)$
δ_i	Informed Search Decay Rate	$E(5)$
ρ_q	Search Quit Probability	$E(30)$
λ_f	Site Fidelity Rate	$U(0, 20)$
λ_p	Reinforcement Rate	$U(0, 20)$
δ_p	Pheromone Decay Rate	$E(10)$
ρ_r	Recruit Probability	$E(10)$
ρ_x	Leave Nest Probability	$E(100)$

Table 1: Ant parameters tuned by the GA. Parameters were initialized randomly using either a uniform distribution (U) or an exponential distribution (E). σ_u and δ_i define turning parameters. ρ_q, ρ_x, and ρ_r are probability rates. λ_f and λ_p are Poisson probability parameters. δ_p is the pheromone decay rate. Parameters extend the model described in Hecker and Moses (2015). As in Figure 2: blue, σ_u, is used in by all three strategies, green parameters are used by the two recruitment strategies, yellow, δ_p, is used by pheromone only, and red parameters are used by nest recruitment only.

- **Nest Recruitment:** Ants probabilistically leave the nest and use random search to look for food. Ants that are not actively searching remain in the nest. As in pheromone recruitment, ants give up and return to the nest with probability ρ_q, and ants that find food pick up the food, survey the area, and if they find food above the threshold, they return directly to the nest and recruit more ants (otherwise they return to the nest and being a new solitary search). The number of ants recruited on a single return to the nest is a fraction of the ants currently in the nest, probabilistic determined by ρ_r. The original ant and the newly recruited ants then return directly to the previous location and perform an informed correlated random walk.

An informed random walk behaves as the uninformed random walk, but with a turning parameter of 4π that decays at a rate δ_i until it reaches σ_u.

These three strategies are well known in the ant literature, but the details can vary among individual species, and in some cases the exact parameters are simply unknown or difficult to measure accurately, such as the nest recruitment rate and the pheromone decay rate. Therefore we use a genetic algorithm (GA) to select each parameter for each environment. The GA-evolved parameters can significantly alter the outcome of a specific strategy. For example, in the pheromone recruitment strategy, if the decay rate of the pheromone is very high, it will dissipate before it is able to be utilized by other ants. This effectively reduces the pheromone recruitment strategy to random search.

Ants begin each simulation in the nest at the center of the grid. Ant behavior is governed by the eight parameters listed in Table 1. Behavior transitions among four possible states: `resting`, `traveling`, `searching`, and `returning` (Fig. 2) at rates determined by the evolved parameters.

To study the effects of different environments on optimal foraging strategies, we model two environmental dimensions: the spatial distribution of food and the volatility of food, where food is a discrete unit, analogous to a seed.

- **Spatial Distribution:** Food is placed randomly in space at one of one, four, or 16 piles, or distributed uniformly (Fig. 1A). These values were chosen to correspond to food distributions in known ant habitats (Hölldobler and Wilson, 1990; Sanders and Gordon, 2002; Markó and Czechowski, 2012). Piles were never placed within 20 grid cells of the nest.

- **Food Volatility:** Food piles are moved at a specified rate to new locations in the grid to simulate growth and decay of resources (Fig. 1B). The volatility rate corresponds to the number of times an ant can make a round trip from the nest to a food pile before the pile has moved. Food volatility was set to be either stable (no volatility), or it was moved at a rate of ten, five or 2.5 round trips. Rates of less than 2.5 round trips eliminated the value of recruitment, while rates above 10 round trips did not show behavioral difference from the stable scenario.

Experimental Design

We use the model to assess the performance of three different strategies across 16 different environments, using a GA to find good parameter values for each strategy/environment pair. This process mimics natural selection, which occurred over evolutionary time scales as each ant species evolved. Having tuned each strategy for a specific environment, we then compare their performance. Each run of the simulation evaluates a single colony of 64 ants foraging on a 200 × 200 two-dimensional grid over one simulated hour. Each cell in the grid represents an 8 × 8 cm patch, so the model simulates a 16 × 16 m area of flat land. The number of ants, spatial extent of the search, and its duration were each based on small desert seed-harvester ant colonies (Flanagan et al., 2012). Selected runs using 320 ants showed results consistent with the main model (data not shown). The simulated ants move through the grid one cell per time-step (Moore neighborhood), foraging for 1,280 food resources (*seeds*) where each cell can contain at most one seed.

We use a generational GA with population size of 25, runs of 50 generations, tournament selection (tournament size of 2), uniform crossover, 10% Gaussian mutation, and elitism, where the single best individual in each generation is retained unchanged. Full details of the algorithm are given in Hecker and Moses (2015). Each individual represents a single ant colony, and individuals are initialized using parameter values drawn from the distribution functions shown in Table 1, column 3. Because the strategies are non-deterministic, each individual's fitness is determined by summing up the number of seeds collected over 8 independent runs of the simulation, where each run lasts for 7,200 time steps. At the end of each GA run, the genomes of the final population of 25 are combined by averaging each gene's values. This combined genome constitutes the resulting colony of the evolutionary run. Finally, 1,000 additional simulations are run with these strategy parameters to assess variance of foraging performance for the strategy/environment pair.

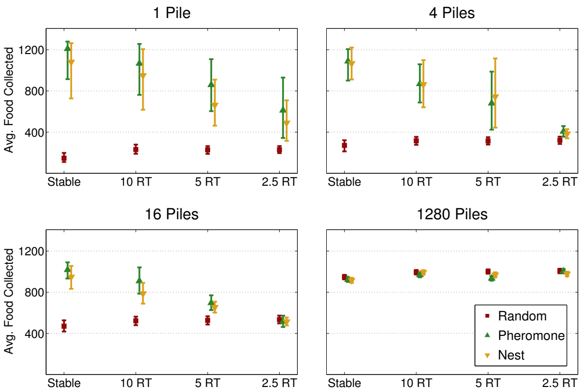


Figure 3: **Search Performance vs. Volatility and Spatial Distribution.** Random, pheromone, and nest recruitment evaluated over 16 environments (four pile counts by four volatility rates). Each bar represents the 95% credible interval resulting from 1,000 runs of the model after an optimal parameter set has been evolved for the specific environment-strategy pair. The two recruitment strategies show decreased performance both as the number of food piles increase and as the food volatility rate increases. The decrease in variance as pile count and volatility increases represents the dominant effect of finding and exploiting clustered piles quickly in the low pile, low volatility environments.

Experimental Results

An optimal parameter set was evolved for each combination of strategy, food distribution, and food volatility. Because the model is stochastic, 1,000 runs of the model were then performed with these fixed parameters to generate 95% credible intervals [1] for each experiment (Fig. 3). Pheromone and nest recruitment perform well when food is stable and arranged in large piles. Random search performs the best in environments with highly dispersed food. This is to be expected as pheromone and nest recruitment leverage information about their environment to improve performance; increasing dispersal and volatility decreases the amount of information gained by finding food. In a majority of cases, nest recruitment performs as well as, or only slightly worse than pheromone recruitment.

The performance of all strategies equalizes when food is uniformly distributed in space (1,280 'piles' of single seeds).

[1] Contains inner 95% of model outcomes.

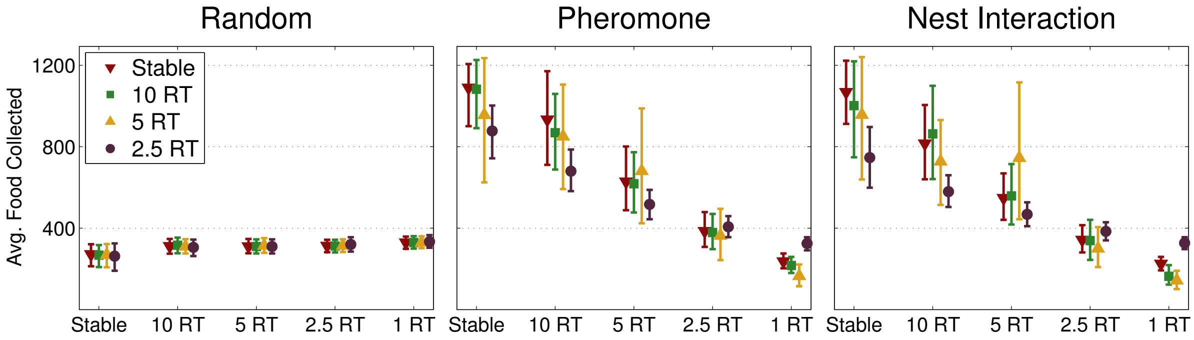


Figure 4: **Fixed Parameters Applied to Other Volatility Rates.** Four pile parameter sets evolved for a specific volatility rate applied to four pile environments with different volatility rates. The consistency of random search illustrates its independence from volatility rate. Pheromone strategy shows little difference between parameter sets evolved for different volatility rates, suggesting a level of robustness. Conversely, nest interaction shows increased performance by strategies evolved for the specific environment, suggesting environment specialization.

In this scenario, the collection of one food item gives no information regarding the location of any other, and the recruitment strategies cannot out-perform random search (as predicted by Flanagan et al. (2011)). Because each strategy uses an optimized parameter set for each experiment, and because random search outperforms both recruitment strategies for 1,280 piles, the recruitment strategies evolve parameters that eliminate information exchange among ants. This explains the similarity of the results in the 1,280 pile environment (Fig. 3). Similarly, information becomes less valuable in highly volatile environments. At volatility rates of 2.5 round trips, recruitment is only effective in the single pile scenario. In these cases of both maximum food piles and high volatility, recruitment strategies evolve away the use of any form of recruitment and behave as random search.

Because random search does not use any memory, food volatility has a minimal effect on its efficiency. Ant colonies using a random walk use similar parameter sets independent of volatility. Conversely, pile distribution does have an effect on random search efficiency. Tight clustering of food resources hinders random search even though total food quantity is held constant. There is also a positive relationship between pile size and overall variance (Fig. 3). Food spread evenly through space results in very consistent searches for all ants. Conversely, the distance from the nest to large piles of food will have a strong effect on the result of any given run. This effect is minimized as volatility increases because high volatility leads to multiple random pile sites.

Evolved values for the correlation in the random walk tend to be small: on the order of 0.1 radians. These values lead to relatively straight search vectors. Because ants return to the nest after finding food, and because their search paths are relatively straight, food hidden behind nearer patches will be found last. The number of straight trajectories from the nest that find food is proportional to the sum of the diameters of the food piles, which scale as the square root of the size of the pile, not counting overlap. This means there are fewer straight trajectories from the nest that intersect food in clustered environments, and may explain why the random search strategy does worse there. Evolved values for the uninformed turning coefficient for the two recruitment strategies are generally higher than those of the random search: on the order of 0.15 radians. A larger turning coefficient corresponds to a random walk closer to the nest. This difference may indicate that the recruitment strategies more thoroughly exploit food resources close to the nest.

Fixed Parameters

Pheromone and nest recruitment are efficient strategies in the 1,280 pile environments and the 2.5 round trip volatility environments only because the optimal parameter sets for each reduce them to random search. We used fixed parameter sets that enforce the use of informed search to better evaluate the recruitment strategies in these environments.

Each strategy evolved unique parameter sets for each level of volatility. Here, parameter sets from specific volatilities were applied to each of the other volatility rates as well as the extremely volatile one round trip case for the four pile food spatial distribution (Fig. 4).

The results show a consistent decrease in performance by the two recruitment strategies as volatility is increased. Recruitment of ants to a localized pile works until the pile moves, at which point ants are recruited to an area that no longer contains food. The fixed parameter strategies are not outperformed by the random strategy in the 2.5 round trip environment, suggesting 2.5 round trips may be an approximate threshold beyond which recruitment does not work. This may be because while highly volatile, persistence of 2.5 round trips still offers ant colonies enough time to leverage information before the pile disappears. Recruitment strategies evolved for more stable environments perform worse than the effectively random search strategy in the extreme

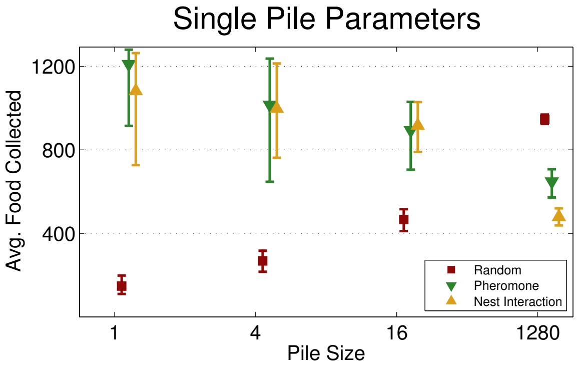


Figure 5: **Fixed Parameters Applied to Other Pile Sizes.** Parameter sets evolved for the stable single pile environment applied to stable environments of all pile distributions. Recruitment strategies designed for a highly clustered environment do poorly as food becomes more distributed. Randomized search strategies evolved for a clustered environment perform well in distributed environments.

volatility case, showing that recruitment strategies can actively hinder search when used in the wrong environment.

Further analysis reveals that pheromone success is not dependent on whether the pheromone parameters were evolved for the proper volatility rate, except in the extreme case: specifically the parameters evolved for the 10 round trip volatility case perform *worse* for that environment than the parameters evolved for the stable case. Conversely, nest interaction performance is always highest by the parameter set evolved for that specific volatility rate. The best example of this in the five round trip environment, where the five round trip strategy improves on the others by over 33% (Fig. 5).

As explained in the previous section, the effectiveness of the random strategy is not strongly correlated with the volatility rate. The results of Figure 4 confirm this and also highlight the increase in performance as volatility increases.

Because each parameter set is evolved for a specific volatility rate, we expect that it should out-perform any parameter set evolved for a different volatility rate. This holds true except for the pheromone 10 round trip data point, where the parameters evolved for the stable environment perform the best. This is likely due to the similarity between the stable and 10 round trip environment, and the inability of the GA to evolve a globally optimal set of parameters for each environment due to computational limitations.

Similar to the fixed volatility parameter runs, results show that recruitment strategies evolved for clustered environments perform poorly in a spatially distributed environment (Fig. 5). In each case, recruitment of other ants to the location of a previously collected food source leads to a now empty area, actively hindering search. Conversely, a random strategy evolved for a clustered environment is able to perform well in more distributed environments as it does not get stuck looking for more food in the same location.

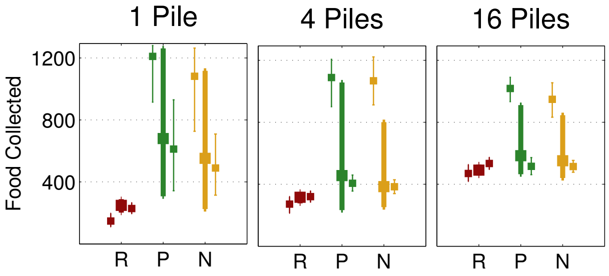


Figure 6: **Search Performance in Random Environments.** Parameter sets were evolved for environments with variable volatility. Results are plotted as mean values inside their 95% credible intervals. Each result (thick middle) is plotted between results for the stable environment (left) and results for the 2.5 round trip environment (right). The results show a slight performance increase versus the 2.5 round trip scenario for the two recruitment strategies. The variance of the results for the recruitment strategies are large, suggesting the use of recruitment in random environments is helpful.

Variable Volatility

A consistent environment is not a reasonable assumption in the real world. To test the ability of each search strategy to cope with an uncertain level of volatility, we evolved new parameter sets in the 1, 4, and 16 pile cases where volatility was chosen uniformly at random to be between 1.25 round trips and stable for each iteration. Assuming 2.5 round trips is a reasonable threshold between environments where recruitment may be used beneficially and not, this produced a distribution of environments where half would benefit from recruitment and half would not (the functional volatility parameter is inversely proportional to the round trip unit). Similar to the original experiment, 1,000 runs were performed (each with a random volatility rate) once the parameters were set by the GA to generate credible intervals (Fig. 6).

Similar to previous results, the success of the random strategy is not strongly affected by volatility. A slight increase in performance versus the stable environment is consistent with results shown in Figures 3 and 4. Pheromone and nest recruitment strategies show slight improvements over the 2.5 round trip environments. Because random search results in low variance, the large variance of the recruitment strategies, as well as the general improvement in performance, shows that the recruitment strategies evolve parameter sets that make use of information, even when that information is short-lived.

Discussion

The fact that ants use different foraging strategies in different environments (Lanan, 2014) suggests that each strategy has been selected and tailored through evolution to perform well in that environment. We used a spatial computational model to study this hypothesis, simulating three general and customizable strategies that subsume most known biological ant behavior. The results show that information-based strategies, such as pheromones and nest recruitment, perform worse as food becomes more spatially distributed and volatile. Success of random search, which does not rely on information, is not affected by food volatility, and performs better when food is widely dispersed. In extreme cases of dispersion and volatility, information-based searches perform worse than random search. Ants foraging in environments with unpredictable volatility are able to improve their performance only slightly using recruitment.

These results are consistent with previous findings (Hecker and Moses, 2015), which considered only food distribution: colonies that forage for clustered resources use recruitment-based strategies to exploit information, while colonies that forage for randomly distributed resources avoid recruiting and instead focus on efficient correlated random search. Further, colonies are most efficient when foraging on the distribution for which they are evolved, although some foraging strategies are sufficiently flexible to function well on different distributions. The study reported here extends this work to consider volatility and suggests that nest recruitment and random search may be better alternatives to pheromone recruitment in the right settings.

Implications For Robot Swarms

Chemical pheromones provide foraging ants with a stigmergic, mass recruitment method that is highly scalable, fully decentralized, and generally tolerant of environments with little or no volatility. Robot swarms that mimic ant pheromones, on the other hand, are restricted to foraging in tightly controlled environments that require complex, monolithic infrastructure. For example, swarm researchers have constructed elaborate stigmergic mechanisms using an always-on ink pen and white paper flooring (Svennebring and Koenig, 2004); a tightly-coupled video camera, video projector, and vision processing system (Garnier et al., 2007); and a phosphorescent-painted floor combined with ultraviolet light emitters (Mayet et al., 2010).

Ants also use simpler, more primitive recruitment strategies such as tandem running and group raids, which include a local recruitment display to stimulate nest mates to return to high-quality food patches (Cassill, 2003). Robot swarms mimic these short-range recruitment strategies using robot-to-robot physical connections (Krieger et al., 2000), nearest-neighbor local communication (Schmickl and Crailsheim, 2008), and robot-chain path formation (Nouyan et al., 2009). These swarms employ relatively simple communication schemes that do not require global coordination or preexisting infrastructure in order to collectively forage for resources or aggregate in target areas.

The results of this study demonstrate that nest recruitment strategies are at least as efficient as pheromone recruitment strategies for many environments. Nest recruitment is relatively simple to implement in robot swarms, while pheromone recruitment requires robot- and environment-specific infrastructure. Further, the foraging success of nest recruiters depends only on local, agent-to-agent communication, while pheromone recruiters often depend on global coordination with a single point of failure. We therefore suggest that research in swarm robotics should focus less on mimicking ant stigmergy, and more on designing and evaluating new decentralized information-sharing protocols that are more scalable and easier to implement in natural environments as foraging strategies for real robots.

Comparison To Biological Ants

Our results show that pheromone and nest recruitment work best in stable clustered environments and that random search works best in environments of high dispersal. We evaluate these statements by comparing them to a comprehensive review of physical ants and their habitats (Lanan, 2014). Lanan categorizes the use of ant search strategies over four environmental dimensions, one being spatial distribution of food and another being frequency of food occurrence, which is similar to volatility. Of the 402 species of ants examined in Lanan (2014), 58 were able to be classified completely into non-overlapping categories.

Of these 58, 13 forage in environments of high food dispersal: seven use random search to forage, three use long term trails. Of the three remaining species, two use a form of nest recruitment in what can be considered moderately volatile, which agrees with our model. The three species that use long term trails forage in a space of high food abundance, such that a trail to a specific location will not exhaust the resources located there. Our model did not explore the effects of high food abundance.

Of the rest of the 58 categorized ant species, 39 forage in environments of high spatial clustering of food. All but four of these use long term pheromone trails, as predicted by our model. Of those, one uses site fidelity in a resource rich area, one is listed as random although the author notes they visit the same location repeatedly, one harvests insects in a highly volatile environment, and one forages randomly and seems to be an exception worthy of future study.

Thus, of the physical ants able to be classified into categories defined by our model, our model immediately agrees with 80.7% of the observations, with an additional 9.6% consistent with the addition of site fidelity and food volatility. interesting case studies for future work.

Conclusions

The phrase ‘ant foraging’ is nearly synonymous with pheromone trails in computer science. However, field studies have shown that numerous ant species do not use pheromone recruitment. This suggests that there are environments for which alternative foraging strategies are at least as efficient as the use of pheromone, or that pheromone based search can be detrimental to the nest.

Lanan (2014) cataloged hundreds of species of ants to create a classification of ant foraging strategies given their environment. Here we analyze three of these foraging strategies across two environmental dimensions: spatial distribution and volatility of food. We find that nest recruitment performs nearly as well as pheromone recruitment in all environments, and that simple random search is more efficient than either when resources are highly dispersed or volatile. Our results, coupled with observations by Lanan, suggest ant species have evolved the use of optimal foraging strategies for their environment.

Understanding how and why ants use different strategies in different environments is critical for biology-inspired algorithmic design. In many cases, an algorithm ill-suited to its environment will perform worse than a simpler naive strategy. Knowing when and how to use these simpler strategies may improve distributed search and swarm robotics.

Acknowledgements

We thank Deborah Gordon, Ben Edwards, and Tatiana Paz-Flanagan for their feedback. This work is supported by NSF grant EF-1038682, DARPA CRASH grant P-1070- 113237, the Air Force Research Laboratory, the Santa Fe Institute, and a James S. McDonnell Foundation Complex Systems Scholar Award

References

Beckers, R., Deneubourg, J. L., Goss, S., and Pasteels, J. M. (1990). Collective decision making through food recruitment. *Insectes Sociaux*, 37:258–267.

Cassill, D. (2003). Rules of supply and demand regulate recruitment to food in an ant society. *Behavioral Ecology and Sociobiology*, 54(5):441–450.

Czaczkes, T. J. (2014). How to not get stuck-Negative feedback due to crowding maintains flexibility in ant foraging. *Journal of Theoretical Biology*, 360:172–180.

Di Caro, G. and Dorigo, M. (1998). AntNet: Distributed Stigmergetic Control for Communications Networks. *Journal of Artificial Intelligence Research*, 9(317-365):317–365.

Dorigo, M., Birattari, M., and Stützle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Mag.*, 1:28–39.

Dorigo, M. and Gambardella, L. M. (1997). Ant colonies for the travelling salesman problem. *Biosystems*, 43(2):73–81.

Evison, S. E. F. et al. (2008). Combined use of pheromone trails and visual landmarks by the common garden ant Lasius niger. *Behavioral Ecology and Sociobiology*, 63:261–267.

Flanagan, T. P. et al. (2011). How ants turn information into food. *Artifical Life*, pages 178–185.

Flanagan, T. P. et al. (2012). Quantifying the effect of colony size and food distribution on harvester ant foraging. *PLoS ONE*.

Franks, N. R. and Richardson, T. (2006). Teaching in tandem-running ants. *Nature*, 439:153.

Garnier, S. et al. (2007). Alice in pheromone land: An experimental setup for the study of ant-like robots. In *Proc. 2007 IEEE Swarm Int. Sym. (SIS 2007)*, pages 37–44, Piscataway, NJ.

Hecker, J. P. and Moses, M. E. (2015). Beyond pheromones: Evolving error-tolerant, flexible, and scalable ant-inspired robot swarms. *Swarm Intelligence*, 9(1):43–70.

Hölldobler, B. and Wilson, E. O. (1990). *The Ants*, volume N1. Belknap Press.

Kacelnik, A. and Bateson, M. (1996). Risky TheoriesThe Effects of Variance on Foraging Decisions. *Integrative and Comparative Biology*, 36:402–434.

Krieger, M. J. B. et al. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406:992–995.

Lanan, M. (2014). Spatiotemporal resource distribution and foraging strategies of ants (Hymenoptera: Formicidae). *Myrmecological News*, 20:53–70.

Letendre, K. and Moses, M. E. (2013). Synergy in Ant Foraging Strategies: Memory and Communication Alone and in Combination. In *Proc. 15th Conf. on Genetic and Evol. Comp.*, GECCO '13, pages 41–48, New York, NY, USA. ACM.

Markó, B. and Czechowski, W. (2012). Space use, foraging success and competitive relationships in Formica cinerea (Hymenoptera Formicidae) on sand dunes in southern Finland. *Ethology Ecology & Evolution*, 24:149–164.

Mayet, R. et al. (2010). Antbots: A feasible visual emulation of pheromone trails for swarm robots. In *Swarm Intelligence: 7th International Conference, ANTS 2010*, volume 6234, pages 84–94, Berlin, DE. Springer Berlin Heidelberg.

Merkle, D. et al. (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, 6(4):333–346.

Moffett, M. W. (1988). Foraging dynamics in the group-hunting myrmicine ant, Pheidologeton diversus. *Journal of Insect Behavior*, 1(3):309–331.

Nouyan, S., Groß, R., Bonani, M., Mondada, F., and Dorigo, M. (2009). Teamwork in self-organized robot colonies. *IEEE Transactions on Evolutionary Computation*, 13(4):695–711.

Pinter-Wollman, N., Gordon, D. M., and Holmes, S. (2012). Nest site and weather affect the personality of harvester ant colonies. *Behavioral Ecology*, 23(5):1022–1029.

Pratt, S. C. (2008). Efficiency and regulation of recruitment during colony emigration by the ant Temnothorax curvispinosus. *Behavioral Ecology and Sociobiology*, 62:1369–1376.

Robinson, E. J. H., Ratnieks, F. L. W., and Holcombe, M. (2008). An agent-based model to investigate the roles of attractive and repellent pheromones in ant decision making during foraging. *Journal of Theoretical Biology*, 255:250–258.

Sanders, N. J. and Gordon, D. M. (2002). Resources and the flexible allocation of work in the desert ant, Aphaenogaster cockerelli. *Insectes Sociaux*, 49:371–379.

Schmickl, T. and Crailsheim, K. (2008). Trophallaxis within a robotic swarm: Bio-inspired communication among robots in a swarm. *Autonomous Robots*, 25(1):171–188.

Schmolke, A. (2009). Benefits of dispersed central-place foraging: an individual-based model of a polydomous ant colony. *The American naturalist*, 173:772–778.

Svennebring, J. and Koenig, S. (2004). Building terrain-covering ant robots: A feasibility study. *Autonomous Robots*, 16(3):313–332.

Theraulaz, G. and Bonabeau, E. (1999). A brief history of stigmergy. *Artificial Life*, 5(2):97–116.

Applying homeostatic neural controller to multi-legged robot and adaptivity to novel disruptions

Hiroyuki Iizuka, Hiroki Nakai and Masahito Yamamoto

Department of Information Science and Technology, Hokkaido University
iizuka@complex.ist.hokudai.ac.jp

Abstract

A robot implemented with a homeostatic neural controller can adapt to disruptions that have not been experienced before. When novel changes occur, the homeostatic neural controller autonomously detects the disruptions from their behaviors and recreates new motions to achieve desired behaviors. In previous studies, the homeostatic neural controller has been only applied to a robot with a simple structure, i.e. wheeled robots which are controlled by adjusting the outputs of wheel motors. It is not clear whether the homeostatic neural controller can work properly when it is applied to a robot with a more complex structure. Therefore, we implement the homeostatic neural controller onto the multi-legged robot and investigate adaptivity. As a result, the robot with the homeostatic neural controller recreates new motion patterns to achieve a desired behaviors after novel disruptions that break a leg.

Introduction

Our human and animals have an ability to adapt to various changes that occur internally and externally. The adaptivity is necessary to survive in the dynamic environment. The adaptation works towards not only the periodic or seasonal changes but also the changes that have not been experienced before. For example, people can adapt to the inverted glasses. Despite the fact that there was no chance to experience the inverted views in the long history of the human evolution, people can recover sensory motor coordination after they wear the inverted glasses for about a week and can catch an object properly or ride a bicycle (e.g., Miyauchi et al., 2004). If the adaptivity can be realized in the robots or software agents, the robots can behave more adaptively in the dynamic environment while achieving their objectives.

In the evolutionary robotics, artificial neural networks are usually used to generate adaptive and robust behavior. In order to obtain proper weight connections for adaptive behaviors evolutionary algorithms are used. It means that the desired behaviors are somehow encoded directly into the static weight connections. The obtained neural networks show robustness toward perturbations using generalization ability of the neural network. However, it is difficult to adapt to the relatively large perturbations like wearing inverted glasses described above because of the direct encoding of desired behaviors into the static weight connections.

A homeostatic neural controller proposed by Di Paolo (2000) is one of the possible implementations that make a robot to adapt to such a kind of disruptions. In his work, the homeostatic neural controller is realized with certain plastic rules and a fitness function to evolve the network to realize generating a desired behavior and maintaining homeostasis at the same time. The weight connections change only when it is required. The study shows that a wheeled robot can adapt to disruptions of exchanging sensor positions like the inverted glasses that have not been experienced before.

Because the homeostatic neural controller can show such an adaptation, there are some extended works to improve the adaptivity (Hoinville and Henaff, 2004; Williams, 2004, Williams, 2005; Iizuka et al., 2013) or to show more cognitive behavior (Iizuka and Di Paolo, 2007; Wood and Di Paolo, 2007). In the most cases of the previous studies, a wheeled robot with simple light sensors is used as an embodied agent and a desired behavior is set to phototaxis or catching an object in order to test adaptivity of the homeostatic neural controller. However, the diversity of possible behaviors becomes very small when the sensors, motors and tasks are very simple. For example, in the case of wheeled robots, the behaviors can be roughly classified into three, i.e., approaching, avoiding or rotating around a light source. For this, any network weight connections result in one of those behaviors. The solution spaces of the neural network for approaching and avoiding are almost same and very large. More simply, when the sensors on the left and right are connected to right and left wheel motors, respectively, the robot can approach a light source. The robot avoids the light source when the sensors and motors on the same sides are strongly connected. Therefore, it is not very difficult to adapt to the disruption of exchanged sensors. It can be overcome by exchanging the connections from sensors to motors somehow. These settings are intentionally kept simple to evaluate adaptivity however it makes unclear if it is applicable to a more complex robot.

There is a study where the homeostatic neural controller is applied to a legged robot (Hoinville and Henaff, 2004). However, the robot only has a single leg. In the similar manner, the adaptation to the disruptions can be realized because of the simplicity of the robot. Therefore, it remains unclear whether the homeostatic neural controller can work properly when it is applied to a robot with a more complex structure. In this paper, the homeostatic neural controller is applied to a multi-legged robot and we investigate how the robot adapts to the different levels of disruptions.

The following sections are organized as follows. First, we describe the homeostatic neural controller and multi-legged robot that we used. Next, the evolutionary algorithm to obtain the parameters of the network is explained. The adaptivity of the multi-legged robot with the homeostatic neural controller is shown in the simulation results.

Homeostatic Adaptation Model

The idea behind homeostatic adaptation is based on the ultrastable system proposed by Ashby (1960). The ultrastable system is a system—open to interaction with the environment—that plastically changes its configuration whenever stability is lost until it finds new internal dynamics which make the system stable under current conditions. Inspired by the idea of ultrastability, the homeostatic neural controller has local plastic mechanisms that change the incoming synaptic weights only when neural activations move out of a bounded region (called homeostatic region) defined in advance. Plasticity continues to operate until the neural activations return to the homeostatic region, resulting in a “stable” configuration in the sense that the network weights do not change further as long as the neural activation remains bounded. When this mechanism is implemented in a simulated robot evolved with a fitness function that simultaneously reward the desired behavior and the maintenance of neural activations within the homeostatic region, the neural controller give rise to the right sensorimotor coordination within a given environmental situation without any synaptic weight changes and the robot behave as desired. If the situation changes, such as an inversion of the visual field or breakdown of body parts, this causes a breakdown of coordination. Under these circumstances, the neural homeostasis of evolved robots also break down. When this happens, the local adaptive mechanism is activated until it find a new synaptic configuration that sustains activations within the homeostatic region. Under these conditions, evolved robots are also able to re-form behavioral coordination (even if they have not been trained to adapt to the induced perturbation).

In this paper, we use the basic homeostatic neural network proposed by Di Paolo (2000). There are different kinds of plastic rules to change the weight connections in the original model but a single rule is used for the simplicity. The homeostatic neural network is applied to a multi-legged robot.

Robot and environment

The robot consists of a main body and 5 legs which have two joints. The schematic view of the robot is shown in Fig. 1. One joint connects the main body with the leg and has a single degree of freedom (DOF). Another joint moves like a knee and also has a single DOF. The total number of DOF becomes 10. Every joint moves in the range $[-\pi/2, \pi/2]$. The robot has the acceleration sensor and can detect the velocity of their movement. The robot is placed on a flat surface of the ground and there is no distraction like walls or objects. Our simulation is executed on PhysX, physics engine (NVIDIA).

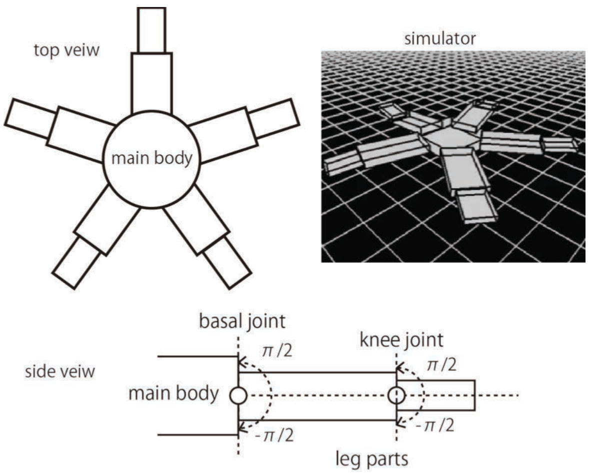


Figure 1: Schematic view of the legged robot. Top left: Top view of the robot which has 5 legs. Bottom: Side view of the leg parts. Each leg has 2 joints which can move in the range $[-\pi/2, \pi/2]$. Top right: The robot implemented on physical simulator.

Homeostatic neural controller

A fully connected continuous-time recurrent neural network (CTRNN) with 11 neurons is used as the robot's controller (Beer, 1990).

The time evolution of the neural states is expressed as:

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{N} w_{ij} z_j(y_j) + I_i, \tag{1}$$

$$z_i(y_i) = \frac{1}{1 + \exp[-(y_i + b_i)]}, \tag{2}$$

where y_i represents the cell potential of neuron i, N is the number of neurons, z_i is the neural activation, b_i is the bias, w_{ij} is the connection weight from neuron j to neuron i, τ_i is its time constant, and I_i represents the sensory input, non-zero only for an input neuron. There is a single input neuron in our neural network. The sensory input is calculated by multiplying a gain parameter (which is set to 3 in this paper) and the average velocity of the main body for last three seconds. There are 10 output neurons to control joints of the robot (2 joints x 5 legs = 10). Each output neuron is assigned to a single joint actuator. The neural activations of the output neurons are linearly mapped to the angle positions of the corresponding joint by the following equation.

$$\theta_i = \pi\left(z_i(y_i) - \frac{1}{2}\right), \tag{3}$$

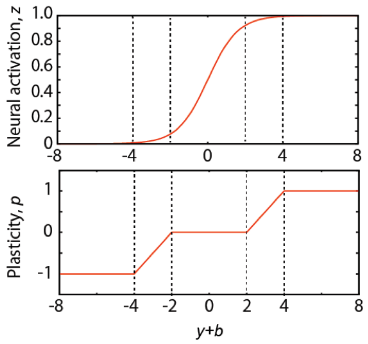


Figure 2: Plasticity as a function of cell potential (bottom). The top graph shows the corresponding neural activations.

where θ_i is the target position of the joint that neuron i controls. Because z_i takes [0, 1], the range of the joint movements is [-$\pi/2, \pi/2$].

The connection weights are initialized by the genetically obtained values at the beginning of a trial while a plastic mechanism allows for the lifetime modification of the connections. A homeostatic region is described as the finite, zero-value set of a plasticity function of the post-synaptic neural cell potential and bias. The homeostatic region is arbitrarily determined to follow the idea of the ultrastable system. The neural states in the most sensitive part of the sigmoid function is defined as the homeostatic region and too high or low neural states are defined as the out of homeostasis. The important thing is that the plasticity function must have two states (working/non-working) to realize intermittent plasticity. Weight w_{ij} from neuron j to i are updated according to:

$$\Delta w_{ij} = \delta_{ij} z_i z_j p(y_i + b_i), \tag{4}$$

where z_i and z_j are the neural activation of post- and pre-synaptic neurons, respectively, Δw_{ij} is the change per unit of time, δ_{ij} is a learning rate (range [-1,1]) that is genetically set for each connection, and $p(y + b)$ is the plastic function that defines the homeostatic region (Fig. 2).

Evolutionary Algorithm

Real-coded genetic algorithm is used to obtain the constant parameters for the homeostatic neural controller. All network parameters, i.e., initial weights w_{ij}, learning rates δ_{ij} , time constant τ_i, and biases b_i, are evolved. They are represented by a real-valued vector ([0,1]) which is decoded linearly to the range corresponding to the parameters (w_{ij}: $[-8,8]$, δ_{ij}: $[-0.9,0.9]$, τ_i: $[0.25, 2.5]$, $b_i[-6, 6]$). Crossover and vector mutation operators, which add a small random vector to the real-valued genotype, are used (Beer, 1996).

The population size in the simulation is 240. The best 12 (5%) agents of the population are kept without changes as the elitism. 60 (25%) agents of the population are generated by randomly mating agents from the previous generation according to rank. The remaining 168 (70%) agents are mutated copies of agents from the previous generation, also selected according to their fitness rank.

The desired behavior for the robot is moving around. The task is to train the desired behavior while keeping the neural activations in the homeostatic regions. The fitness function evaluates how much the desired behavior is achieved (f_d) and how much the neural activations stay in the homeostatic regions (f_h) at the same time. Those two evaluations are multiplied. The actual fitness function is as follows.

$$F = f_d \times f_h \quad , \tag{5}$$

$$f_d = \begin{cases} \dfrac{|p_e - p_i|}{12} & \text{if } d_e \le 12 \\ 1 & \text{otherwise} \end{cases} \quad , \tag{6}$$

$$f_h = \frac{1}{(T-10) \times N} \sum_{t=10}^{T} \sum_{i}^{N} h(z_i^t) \quad , \tag{7}$$

$$h(z) = \begin{cases} 1 & \text{if } \; 0.119 \le z \le 0.881 \\ 0 & \text{otherwise} \end{cases} \quad , \tag{8}$$

where p_i and p_d represent the positions at the beginning and the end of a trial, N is the number of neurons, T is time for the trial, and $h(z)$ is a function that check if the neural activations are within the homeostatic regions. At the beginning of a trial, all the genetic parameters are set to the homeostatic neural network for the robot. The robot can move around based on the outputs of homeostatic neural controller for T (=40 in this paper) seconds as a single trial. f_d simply measures the distance from the initial to the final position. f_h calculates the time-average of the proportion of neurons that have behaved homeostatically. Because the behavior is not stable for the first 10 seconds, the last 30 seconds out of 40 seconds are used for the evaluation of the homeostasis. The fitness function is obtained by multiplying f_d and f_h. By this, we can obtain the robot that can move around while keeping the neural activations in the homeostatic regions.

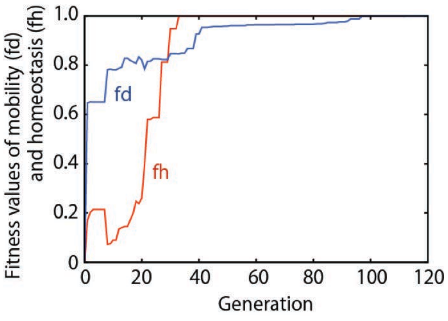


Figure 3: Fitness values of mobility (f_d) and homeostasis (f_h) during the evolution.

Simulation Result

The robot implemented with the homeostatic neural controller could achieve the task, i.e. moving around while keeping the neural activations in homeostatic regions. Figure 3 show the fitness values of mobility and homeostasis during the evolution. The further investigation for adaptivity is performed with the best evolved robot obtained at the 120 generation.

Behaviors for the longer time span

During the evolution, each robot is tested for 40 seconds and the generated neural dynamics and behaviors are evaluated. The fitness values of the mobility and homeostasis reach the maximum value, which means that the robot can move around while keeping homeostasis for 40 seconds. However, it is not clear if the robot can behave properly for more than 40 seconds.

The robot is tested for 300 seconds to see the stability of the behaviors. Figure 4 shows the average velocity of the main body of the robot for 300 seconds. The robot can keep moving around even after 40 seconds while keeping the homeostasis. This result shows that the evolved robot has enough stability to maintain the structure and to generate the desired behavior.

Robustness against relatively small disruptions

In order to investigate adaptivity of the obtained homeostatic neural controller, a disruption which the robot has never experienced during the evolution is introduced. In the experiment, one of the knees gets stuck and the angle of the knee is fixated. The knee is the second joint from the body. Because the broken knee is just fixated at a certain angle and it is still available as a leg by moving the basal joint, the disruption is not large in terms of moving around.

The left graph in Fig. 5 shows the average moving velocity. For the first 40 seconds, the whole body parts move properly based on the neural outputs, however a single knee is broken at time 40. The neural output that controls the knee is just ignored after that. Before time 40, the robot can move around with about 0.7 m/s. After the disruption happens, the velocity gradually decreases. However, the disruption does not cause the serious damage to the robot. The velocity converges around 0.5 m/s and is stabilized. The right graph in Fig. 5 shows the neural activations of a selected neuron in the homeostatic neural controller. Before the disruption, the neural activation oscillates in the homeostatic region and keeps the neural structure (weights) constant except for the beginning of the simulation. After the disruption, the neural activation changes very slowly without moving out of the hostatic region. It means that the neural structure (weights) does not change because the plasticity mechanism does not work within the homeostatic region. The neural activations change in response to the new sensori-motor coordination formed under the knee-broken situation. The adaptivity toward the disruption is realized through robustness, which is the generalization ability as a usual neural network.

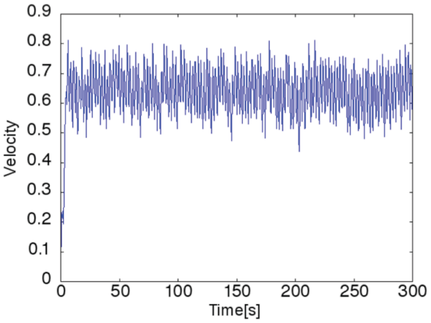


Figure 4: Average velocity of the robot that behaves for 300 seconds.

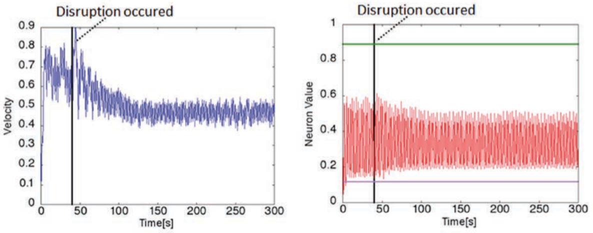


Figure 5: Left: Velocity of the robot. Right: Neural activation of a selected node. The green and pink lines shows the maximum and minimum values of homeostatic regions, respectively. A single knee joint is broken at time 40.

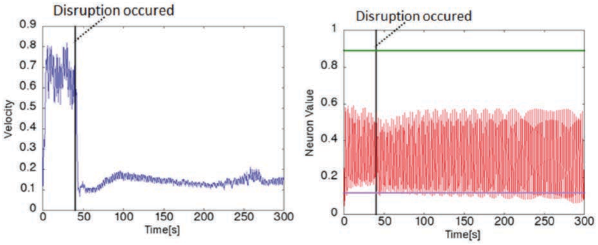


Figure 6: Average velocities of the robot (left) and neural activations (right). The two joints of the same leg get stuck at 40. The green and pink lines shows the maximum and minimum values of homeostatic regions, respectively.

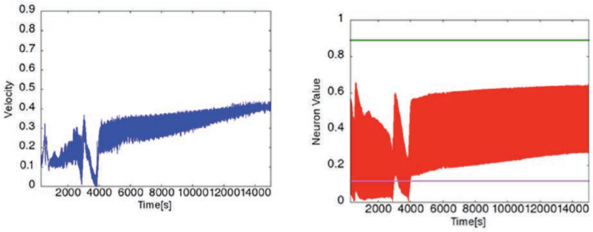


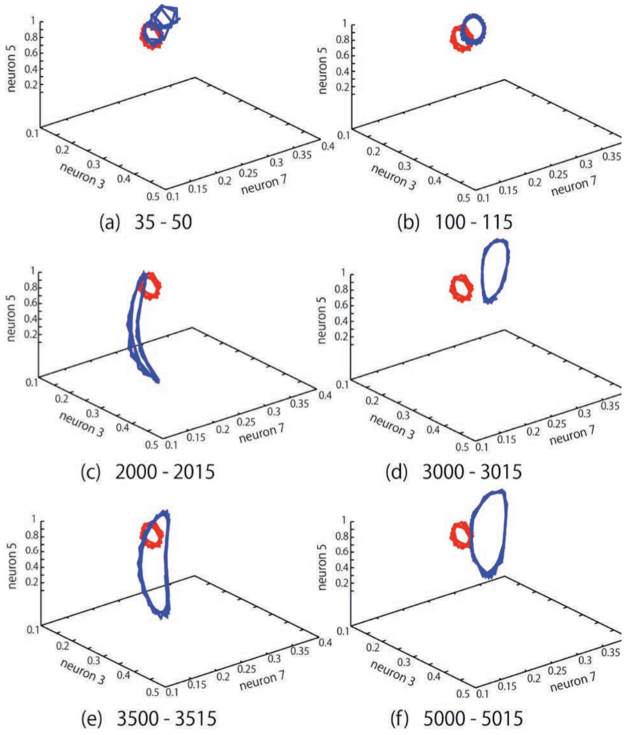
Figure 7: Continuation of average velocities of the robot (left) and neural activations of a selected neuron (right) of Fig. 6. These are the results of the same experiment shown in Fig. 6 but they are shown in longer time span.

Homeostatic adaptation against large disruptions

A relatively large disruption is introduced to investigate adaptivity of the homeostatic neural controller against the more serious damaged situation which the robot also has never experienced during the evolution. The single knee is broken in the previous section, however the knee and the basal joints of the same leg are broken at the same time here. It means that the first and second joint of the leg are fixated at the same time. Figure 6 shows the average velocity and the neural activations from 0 to 300 seconds. The disruptions happen at time 40 same as the previous section. Soon after the disruptions, the robot gets stuck and the moving velocity becomes very slow. The collapse of the desired behavior suddenly causes the change of the input to the homeostatic neural controller and the neural activation cannot be kept in the homeostatic regions due to the sudden changes of input. Then, the plasticity mechanism starts working and changes the network structure (weights). The plasticity keeps working until the network finds the new homeostatically stable states. Figure 7 shows the continuation of average velocity and the neural activations of Fig. 6 to see the adaptation by the plasticity mechanism in the much longer time scale. It shows that the robot behavior is recovered very slowly compared with the time scale in the previous section. The moving velocity returns to about 0.4 m/s. It is not full recovery because the velocity is around 0.7 m/s before the disruptions. However, the neural activations also start returning to the inside of the homeostatic regions at the same time. Therefore, the neural structure can be kept after the activation returns. Consequently, the robot keeps moving around after the homeostatic adaptation.

During the homeostatic adaptation processes, internal representations of sensori-motor coordination must change. Figure 8 shows the neural states of homeostatic neural network in the different phases. The red lines in all graphs indicate the original neural states when they are formed without any disruptions and blue ones show the changes of neural states after the disruptions happen. Before the disruptions, the neural states changes periodically and form the limit cycle to generate stable moving-around behaviors. After the disruptions happen, the limit cycle is collapsed (Fig. 8(a)). The neural states apparently find a similar limit cycle close to the original one (Fig. 8(b)). However, the structure is gradually changing and cannot be fixed because the neuron values do not stay in the homeostatic regions (Fig. 8(b), (c), (d) and (e)). After time 5000, the neural states can keep them within the homeostatic regions and repeat the same neural changes (Fig. 8(f)). At that time, the behaviors are also sufficiently good and the behavior can be repeated with the new sensori-motor coordination.

The motion patterns are also shown in Fig. 9. The dynamics is not as simple as the neural activations because the body interacts with the environment. After the disruption occurs, the motion behavior breaks down ((a) and (b)), which leads to the slowdown of the movements. The behavior dynamics form a different attractor when the movement of the robot become close to zero ((d) and (e)). After recovering the movements, the behavior is stabilized and kept (f).



(a) 35 - 50

(b) 100 - 115

(c) 2000 - 2015

(d) 3000 - 3015

(e) 3500 - 3515

(f) 5000 - 5015

Figure 8: Neural activations of selected three neurons. The data is plotted during different period shown at the bottom of each graph. Time is corresponding to time in Fig. 7. The red lines in all graphs indicate the neural states from 20 to 35 where the disruption is not introduced yet.

To show how much the homeostatic adaptation contributes to the behavior recovery, the plasticity mechanism is removed when the disruptions happen and we see how much the behavior can be recovered from the serious damaged situation without any neural structural changes. This is adaptation without the plasticity mechanism, that is, only robustness works. The velocity after the disruptions is about 0.06 m/s. Compared with the velocity when the plasticity mechanism works, it is much lower. The velocity is not improved without the plasticity mechanism. It means that the behavior recovery for the large disruptions is realized by the homeostatic adaptation.

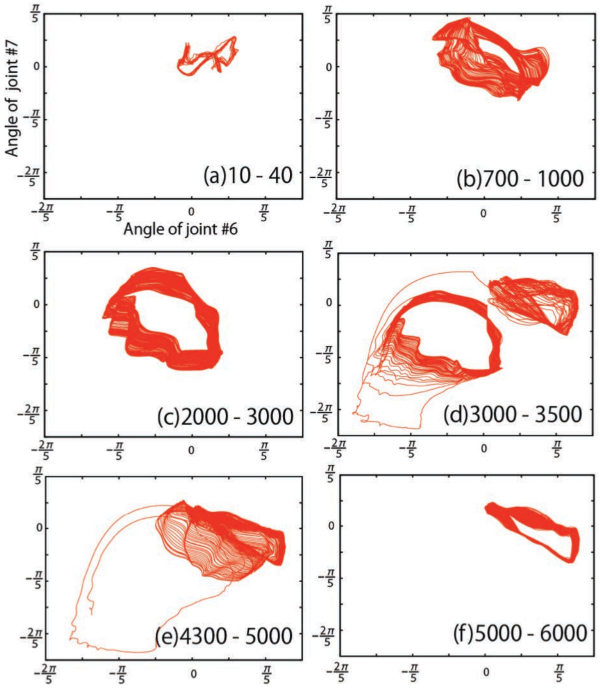


Figure 9: Motion patterns of selected two joints on the same leg. Time shown in the right bottom in each graph corresponds to Fig. 7.

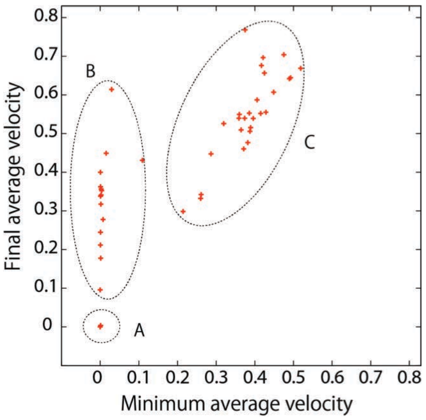


Figure 10: Minimum and final average velocities after two joints get stuck. A dot shows a pair of joints. There are ten joints in the robot. The all 45 pairs are plotted.

Joint	0	1	2	3	4	5	6	7	8	9	any
A	1	0	0	0	0	1	1	0	2	1	3/45
B	8	1	1	3	2	1	5	3	4	2	15/45
C	0	8	8	6	7	7	3	6	3	6	27/45

Table 1: Number of occurrences in which a specific joint appears in a pair classified into A, B, or C group. The total number of pairs is 45. The even joint number indicates a joint connecting the main body with a leg and odd number is assigned to a knee joint.

Not always possible to adapt

The previous sections show the processes of adaptation to the disruptions when it is successfully adapted. The homeostatic neural controller cannot always adapt to the disruptions. As our real life is, if the changes are too large or critical, the robot cannot recover the desired behaviors. In fact, there are some cases that the robot just stop moving when different pairs of joints get stuck in our simulation. The results of the all cases of different pairs of joints are shown in Fig. 10. The graph indicates the minimum and final average velocities after a pair of joints get stuck. The final average velocities are calculated over 1000 seconds after time passed long enough since disruption happens. The results are classified into three groups. The first group is that the robot becomes unable to move once and stops in the end (Fig. 10(A)). The second is that the robot becomes unable to move once however recovers the movements (Fig. 10(B)). The last group is that the final and minimum velocities are almost same and the robot keeps moving before and after the disruption (Fig. 10(C)). The group (A), (B), and (C) correspond to no adaptation, homeostatic adaptation, and robustness. The robustness and homeostatic adaptation accommodate 60% and 33% of disruptions, respectively. In the remaining 7% cases, the robot cannot adapt.

Discussion

Adaptation happens in the following manner. To realize the desired behavior (i.e. moving around) is associated with the internal homeostasis by the evolutionary algorithm. Because the input of the homeostatic neural network is the velocity of the main body, the input becomes high when the desired behavior is achieved. It means that the input values are always high when the internal homeostasis is realized. In other words, to maintain the homeostasis requires the high input values. When the disruptions occur and the behavior is damaged, the velocity becomes slow and the input value decreases. If the robustness cannot accommodate the changes, it causes the breakdown of the internal homeostasis. The plasticity starts working because of the breakdown of the homeostasis until it finds a new structure that can get enough input values to maintain homeostasis, which leads to the desired behavior.

Our results show that there are possibilities that the multi-legged robot implemented with the homeostatic neural controller can recover the desired behavior when novel disruptions happen. In the homeostatic regions, the neural dynamics also show robustness to the disruptions. If it is not accommodated by the robustness, the homeostatic neural controller start changing network structures. The multi-legged robot have the robustness and homeostatic adaptation at the same time.

However, the recovery by the homeostatic adaptation were sometimes not good enough and sometimes no adaptation occurred. We discuss about improvements. Firstly, regarding to the plasticity mechanism, the method that we used here is based on Di Paolo's homeostatic adaptation model and Hebbian type of learning is used. Although the parameters of the methods were obtained by the genetic algorithm, the selection pressure for these parameters must be weak because those parameters work only when the internal homeostasis breaks down. It may be useful to give a specific serious damage to the robot during evolution in order to train these parameters and to adapt to the different disruptions that have not been experienced. This is also related to the learning speed. In fact, the learning speed is very slow in the current setting. To increase the learning rates would improve speed however it is connected with the whole adaptation dynamics as explained in this paper. It should be self-organized during evolution.

Secondly, the relation between the input values and the desired behavior is rather direct. The direct relation makes adaptivity poor. The advantage of the homeostatic neural controller is that the whole mechanism including the evaluation of the desired behavior is distributed in the network. Nothing is centralized. The learning mechanisms (e.g. backpropagation or genetic algorithm) can be applied to adjust the weight connections with the input signal instead of using homeostatic mechanisms. However, if the learning and evaluation mechanism has a damage, the system will not work. In the case of the homeostatic mechanism, those mechanisms are not centralized. From this point of view, the input value of the current setting is not enough. When the sensor is broken to measure the velocity, the system will not work properly. This should be improved for the multi-legged robot.

Conclusion

In the conventional studies, the homeostatic adaptation was only applied to a wheeled or simple robot but we implemented the homeostatic adaptation model on the multi-legged robot. In the same manner as the conventional method, the homeostatic neural network was evolved to perform a desired behavior while keeping the neural states in the homeostatic regions. Because we used more complex structured robot, different levels of disruptions could be considered. When the small disruption was introduced, the robot with the homeostatic neural network kept moving around without any structured changes of the network. However, the homeostatic adaptations happened when the relatively big disruptions were given and the robot could recover the desired behavior. Our result shows that the homeostatic neural network can work properly even in the more complex robot and the problems of applying homeostatic adaptation was discussed.

References

Ashby, W. R. (1960). Design for a brain: The origin of adaptive behavior (2nd ed.). London: Chapman and Hall.

Beer, R. D. (1990). Intelligence as adaptive behavior: An experiment in computational neuroscience. San Diego: Academic Press.

Beer, R. D. (1996). Toward the evolution of dynamical neural networks for minimally cognitive behavior. In P. Maes, M. J. Mataric, J.-A. Meyer, J. B. Pollack, & S. W. Wilson (Eds.), From Animals to Animats 4: Proceedings of the 4th International Conference on Simulation of Adaptive Behavior (pp.421-429). Cambridge, MA: MIT Press.

Hoinville, T. and Henaff, P. (2004). Evolving plastic neural controllers stabilized by homeostatic mechanisms for adaptation to a perturbation. Proceedings of the 9th International Conference on the Simulation and Synthesis of Living Systems, 81-87.

Iizuka, H. and Di Paolo, E. A. (2007). Toward Spinozist robotics: Exploring the minimal dynamics of behavioural preference. Adaptive Behavior, Volume 15, No.4, 359-376.

Iizuka H., Ando H. & Maeda T. (2013) Extended homeostatic adaptation model with metabolic causation in plasticity mechanism‐ Toward constructing a dynamic neural network model for mental imagery. Adaptive Behavior 21(4): 263–273.

Miyauchi, S., Egusa, H., Amagase, M., Sekiyama, K., Imaruoka, T., Tashiro, T., (2004). Adaptation to left–right reversed vision rapidly activates ipsilateral visual cortex in humans. Journal of Physiology-Paris 98 (1-3): 207–19.

NVIDIA, PhysXSDK, https://developer.nvidia.com/physx-sdk

Paolo, E. A. (2000). Homeostatic adaptation to inversion in the visual field and other sensorimotor disruptions. In Meyer, J., Berthoz, A., Floreano, D., Roitblat, H., & Wilson, S. (Eds.), From Animals to Animats VI: Proceedings of the 6th International Conference on Simulation of Adaptive Behavior, 440-449. Cambridge, MA: MIT Press.

Williams, H. (2004). Homeostatic plasticity in recurrent neural networks. From Animals to Animats 8: Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior: 344-353.

Williams, H. (2005). Homeostatic plasticity improves continuous-time recurrent neural networks as a behavioural substrate. In Proceedings of the 3rd International Symposium on Adaptive Motions in Animals and Machines.

Wood, R. and Di Paolo, E. A. (2007). New models for old questions: Evolutionary robotics and the 'A Not B' error. Proceedings of 9th European Conference on Artificial Life:1141-1150.

Investigating a cellular automata model that performs three distance diffusion on a robot path planning

Gina M. B. Oliveira[1], Patrícia A. Vargas[2] and Giordano B. S. Ferreira[1,3]

[1]Universidade Federal de Uberlândia, Uberlândia, Brazil
[2]Heriot-Watt University, Edinburgh, United Kingdom
[3]Tufts University, Medford, MA, United States
gina@facom.ufu.br

Abstract

An improved cellular automata (CA) model is proposed and evaluated on a path planning problem for an autonomous robot. The objective is to construct a collision-free path from the robot's initial position to the goal by applying the improved CA model and environment pre-processed images captured during its navigation. CA rules are used to enlarge obstacles and to perform three distance diffusion. Goal distance is spread by a CA rule using the free cells. Distance diffusion spread two new metrics used for cells inside the enlarged obstacle regions. The new distances are then used to plan routes when the robot needs to pass inside enlarged obstacle regions. The algorithm performs a new path planning at each n steps of robot navigation using its current position. Our inspiration came from the possibility to mimic the cognitive behaviour of desert ants, which re-plan their path to the goal constantly in time intervals, using only local cues. An e-puck robot was used in simulated scenarios to evaluate the new CA based adaptive path planning model. The simulations show promising results on the single robot's performance confirming that the model could also be adapted for robot swarms.

Introduction

Path planning is one of the most studied tasks for autonomous mobile robots (Arkin, 1998) (Siegwart et al., 2011). The objective is to define a list of movements from an initial position to the goal. There is a lot of research on using for example "A^* like" techniques for path-planning (Hernández et al., 2011) (Phillips et al., 2014). Cellular Automata (CA) are totally discrete models and have been recently considered for path planning Behring et al. (2000), (Parker et al., 2003), (Ioannidis et al., 2011), (Ferreira et al., 2014). CA consist of a large number of simple components with local connectivity. Despite of the simplicity of their basic components, CA are able to solve very complex problems such as scheduling (Swiecicka et al., 2006) and cryptography (Oliveira et al., 2010).

This work proposes an improved CA model derived from (Behring et al., 2000) which was further explored in (Parker et al., 2003), (Tavakoli et al., 2008) and (Kostavelis et al., 2012). The results show investigations of the new model for a single robot. It is important to indicate that our main objective is not to propose a better method for path planning by comparing, for example, with A^* like algorithms (Hernández et al., 2011) (Phillips et al., 2014). Here the CA model is used to further investigate its intrinsic distributed feature performance in this particular robot application. Our inspiration came from the possibility to mimic the cognitive behaviour of desert ants using CA. These ants re-plan their path to the goal constantly in time intervals, using only local cues as there is no pheromone to guide them (Tinbergen, 1932) (Wystrach et al., 2014) (Collett et al., 2014). This work is a first step towards fully exploiting CA intrinsic parallelism.

The planning algorithm proposed here uses a CA to enlarge the obstacle cells initially identified by an environment pre-processed image that generates a grid map. It is beyond the scope of this paper to give a full account of the pre-processing image system. The reader should refer to (Behring et al., 2000) for further details.

The distance to goal (GD) is calculated by CA iterations for all the free cells starting from those closest to the goal up to the initial position. The major enhancements in the present work are: (i) the path planning is continuously applied while the robot is navigating at each n time steps; (ii) a new distance diffusion was incorporated to the model in which two new metrics are calculated for the enlarged obstacle cells, i.e. the *Next Free Cell Distance* (NFCD) and the *Next True Obstacle Distance* (NTOD). The first modification makes it possible to approximate the actual path performed by the robot to the ideal route calculated in the initial position, turning its final position closest to the goal. The second modification is performed by two new CA rules to diffuse the new distances, which are used to help the robot to escape from regions near to obstacles. The V-REP simulator (Robotics, 2015) was used to evaluate the new model through experimenting with a single e-puck robot (EPFL, 2015).

This paper is structured as follows: Section 2 reviews CA to path planning. The model proposed by Behring and collaborators (2000) is described in Section 3. Section 4 describes the new model and shows the simulation results. Section 5 concludes the paper and suggests future work.

Cellular automata and path planning

Cellular automata are totally discrete dynamical systems composed by simple components with local interactions. Basically, a cellular automaton consists of two parts: the cellular space and the transition rule. Cellular space is a regular *d*-dimensional lattice of *N* cells, each one with an identical pattern of local connections to other cells. CA are characterized by a transition rule that determines which will be the next configuration of the lattice, from its current one. Cells interact locally in a discrete time *t*: the state of the cell *i* at time *t + 1* depends only on the states of its neighborhood at time *t* including cell *i*. Cells updating is usually performed in a synchronous way. Considering 2D lattices, spatial neighborhoods must be used as the Moore neighborhood, which is composed by the central cell and its eight immediate neighbors (Sarkar, 2000).

CA are able to represent high complex phenomena at the same time that they can be exactly simulated by digital processors due to its intrinsic discrete nature. Moreover, CA-based models can be efficiently executed by multiprocessors architectures due to its inherent parallelism. Finally, the decentralized CA architecture permits the development of high-distributed solutions (Swiecicka et al., 2006), (Oliveira et al., 2010). Those characteristics lead CA to be considered for robotics, more specifically for path planning (Shu and Buxton, 1995), (Marchese, 2002), (Behring et al., 2000), (Ioannidis et al., 2011), (Akbarimajd and Hassanzadeh, 2012) and (Ferreira et al., 2014). A classification scheme was presented in (Ferreira et al., 2014) which divides the previous works on CA applied to path planning into six approaches. In this work we will concentrate on investigating and improving a previous model from the distance diffusion planning approach.

Distance diffusion planning

(Behring et al., 2000) presented an approach that is one of the most cited path planning models using CA. The robot is considered as a single non-oriented point not subjected to kinematics and dynamic laws. The space is a discrete 2D space divided into square cells in such way that the robot occupies a unique cell. In a pre-processing phase, the algorithm receives an environment pre-processed image as input with obstacle positions, the robot initial position and the goal cell. The discrete environment is transformed into a CA lattice cell and each cell state has four possible values - Free, Obstacle, Initial and Goal.

The planning algorithm is divided into three phases. The first phase provokes the enlargement of the obstacles initially identified aiming not only at compensating the effects of the dynamics not considered in the model but to avoid obstacle collisions during the robot navigation. The CA transition rule that enlarges the obstacles is defined by:

Rule Enlargement:

If the cell is *Free* and one of its neighbors is *Obstacle*, **then** the next value of the central cell is *Obstacle*;

This rule is applied for a fixed number of time steps (x), which depend on the cellular space resolution and robot size. Behring and colleagues used $x = 4$, which results in an enlargement of 4 cells in obstacles' thickness.

A CA rule on the second phase spreads the Goal Distance (GD) for each free cell. Free cells states change to integer numbers, which are the distance from the cell to the goal in terms of robot's steps. The goal cell changes its state to 0. The CA rule only updates free cells states and is defined by

Rule Goal Distance:

If the central cell is *Free* and one of its neighbors is v, **then** the next value of the central cell is $v + 1$;

Applying this CA rule for some steps, GD is calculated for all free cells spreading from the goal to the robot's neighbors, if there is a collision-free route from the initial cell to the goal. If this route does not exist, the spreading will be stopped before the initial cell is reached. Figure 1 shows an example of goal distance spreading. The last phase uses GD values to construct a collision free route starting from the initial position of the robot until the goal cell, decrementing 1 in the value of GD at each step. Figure 1.b shows an example of a planned path.

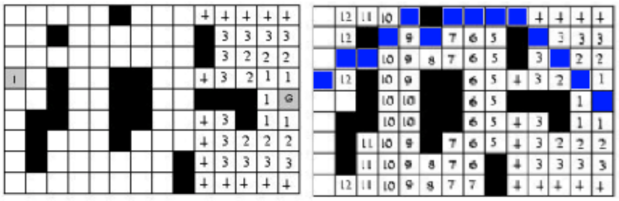


Figure 1: GD diffusion and planned route adapted from (Behring et al., 2000)

Adaptive path planning model

Simulations showed a significant difference between the planned route and the actual one performed by the e-puck robot using the original model described in (Behring et al., 2000). An example of such situation is presented in Figure 2.a, which indicates the route planned in the start of simulation (in green) and the actual path traveled over by the e-puck (in red).

The original model was modified to overcome this first critical point, in a way that the algorithm would now perform new distance diffusion at each n time steps. Using this strategy, even if the robot starts to diverge from the expected route, a new path planning will be executed, correcting the deviation. For such, the method needs to receive a pre-processed new image at each n time steps to restart the path planning using the current robot position. An adequate value for n can be determined according to the latency of the image processing system. All simulations were carried out using $n = 5$. Figure 2.b shows the planned path and actual one performed using the distance diffusion at each 5 steps for the same initial scenario of Figure 2.a. One can see that although the robot's route performed with the adaptive planning is not similar to the original planned path, after correcting its route, the robot's final position is very close to the planned goal. This modification turns the new model into an adaptive path planning, instead of a static planner as (Behring et al., 2000).

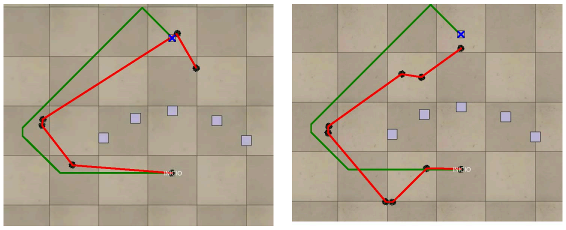


Figure 2: Planned route (in green) and the observed navigation (in red): (a) Behring et al.s model (b) New adaptive planning model.

The second critical point identified in the original model refers to situations in which the algorithm cannot plan a path because the robot is initially positioned inside enlarged obstacle cells. The main reason why the algorithm in (Behring et al., 2000) cannot plan a valid path in this case is because all enlarged obstacle cells are treated as true obstacles and the CA diffusion rule is not applied to them in the second phase. Therefore, GD spreading does not achieve the initial position and the path cannot be constructed. This situation can happen quite often from the initial scenario, i.e. when the plan is applied for the first time. However, this kind of case is more likely to occur when using the new adaptive path planning at each n steps. This is due to the fact that several times the original route passes in the limit of enlarged regions and during navigation the robot invades such areas.

In order to overcome this second critical point we first considered applying GD spreading also in the enlarged obstacle cells, using higher values to these cells to avoid using them by the route planner. However, if the distance to the goal is also used in these regions, when the routes needed to be calculated inside them, the planned route must approximate to the true obstacles, increasing the risk of collisions if they make the robot closer to the goal. Consequently GD metric was discarded as it cannot be used in such regions.

Two new measurements were then calculated using CA rules and both were applied only to obstacle cells. The first indicates the shortest distance of an obstacle cell to a true obstacle, i.e. Next True-Obstacle Distance (NTOD). The second measurement indicates the shortest distance of an obstacle cell to a free one and it is calculated only for enlarged obstacle cells. It starts from a relatively high limit (V) and it increments in one unit as the distance also increments. We call it the Next Free-Cell Distance (NFCD). Both metrics are used to construct a route that takes the robot to free cells as fast as possible, but also takes into account the need to pass as distant as possible from the real obstacles.

Each cell state is composed by a triple of sub-states $(s1, s2, s3)$: **s1** is a categorical sub-state with four possible values - Free, Obstacle, Initial and Goal; **s2** is a numeric sub-state that store the distance between a cell and a true obstacle; **s3** is a numeric sub-state that can store the distance between a cell and the goal (if the cell is free) or the distance between a cell and the free one (if the cell is an enlarged obstacle). Note that sub-states $s2$ and $s3$ are finite values which maximum values depend on environment grid dimensions (NM cells) being that the maximum possible values are N+M (the maximum Manhattan distance). Moreover, this CA model uses a fixed boundary condition, since the extreme/border cells of the 2-dimensional lattices is considered to have obstacle cells (the wall) on all their non-defined neighbors.

The planning algorithm is applied at each n steps of the robot locomotion and it is divided in a pre-processing and four major phases. It uses CA rules with Moore neighborhood (Sarkar, 2000) in the first three phases. In the pre-processing phase, the algorithm receives an environment pre-processed image as input, the robot initial position and the goal cell. It then establishes the initial value of s1 for each cell of the lattice. The new proposed algorithm is described as follows.

Phase 1 - causes the enlargement of the obstacles initially identified and spreads the value of next true-obstacle distance (NTOD) at same time. NTOD is stored at sub-state $s2$. It starts establishing that $s2 = 0$ for all obstacle cells by applying the following rule one:

Rule NTOD Initialization:

If the central cell has $\{s1 = Obstacle\}$, **then** the next value of the central cell is $s2 = 0$.

The CA rule transition enlarges the obstacles and updates the values of sub-states $s1$ and $s2$ and it is defined by:

Rule Enlargement and NTOD Diffusion:

If the central cell has $\{s1 = Free$ and one of its neighbors has $s1 = Obstacle$ with $s2 = v\}$, **then** the next values of $s1$

and $s2$ of the central cell are *Obstacle* and $v+1$, respectively.

This rule is applied for a fixed number of time steps (x), which results in an enlargement of x cells in obstacles' thickness and in the calculus of NTOD for these cells. The behavior of this CA rule can be observed in Figure 3.

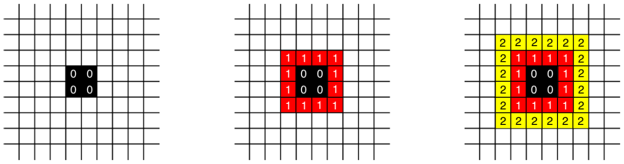


Figure 3: Obstacle enlargement and NTOD diffusion (x = 2).

Phase 2 - uses a CA rule to spread the goal distance (GD) into free cells, as in (Behring et al., 2000), except that now they are stored in sub-state $s3$. At the start of second phase we have that all free cells, obstacle cells and the cell with the robot ($s1 = Free$ or $s1 = Initial$ or $s1 = Obstacle$) have $s3$ undefined. The initialization rule is applied once and it establishes that the goal cell will assume $s3 = 0$:

Rule GD Initialization:

If the central cell has $\{s1 = Goal\}$, **then** the next value of the central cell is $s3 = 0$.

The transition rule updates $s3$ value and it is applied only to free cells by some time steps until the state of $s3$ is not modified for any cell in the lattice. The CA rule transition is defined by:

Rule GD Spread:

If the central cell has $\{s1 = Free$ and $s3$ is undefined and one of its neighbors has $s3 = v\}$, **then** the next state of the central cell is $s3 = v + 1$.

Applying this CA rule for some steps, GD (stored in $s3$) is calculated for all free cells. The resultant calculus is the same illustrated in Figure 1, except that is the new model it represents the $s3$ spreading.

Phase 3 - uses a CA rule to spread the next free-cell distance (NFCD) through obstacle cells, which will also be stored in sub-state $s3$. This phase will only be applied if at the end of the second phase the value of $s3$ is still undefined for neighbor cells of the robot's initial position. In other words, it is applied when the GD spreading finished but the initial position was not reached. In this case, it is probable that the robot is inside an enlarged obstacle area or there is a barrier with enlarged obstacle cells between the goal and the initial position. As a result, the value of next free-cell distance will be calculated for the enlarged obstacle cells and it will be stored in sub-sate $s3$ of such cells. At the end of the second phase, all obstacle cells have undefined $s3$. An initialization rule is applied to start the value of the obstacle cells closest to free cells. This rule is applied to all lattice cells:

Rule NFCD Initialization:

If the central cell has $\{s1 = Obstacle$ and one of its neighbors has $s1 = Free\}$, **then** the next value of the central cell is $s3 = V$.

V is an arbitrary discrete value chosen to be bigger than any goal distance calculated to free cells. For instance, we chose $V = 100$ because it is impossible to have $GD > 99$ due to the lattice dimensions used in our experiments. After using this rule to initialize $s3$ values for the obstacle cells in the border of the enlarged regions, the algorithm applies a CA rule to spread next free-cells distance (NFCD). This rule transition is defined by:

Rule NFCD Spread:

If the central cell has $\{s1 = Obstacle$ and $s2 > 0$ and $s3$ is undefined and at least one neighbor has $s3$ defined$\}$, **then** the next value of the central cell is $s3 = vmin + 1$, where $vmin$ is the minimum value of $s3$, considering all the neighbors of the central cell.

Applying this CA rule for some steps, the NFCD stored in $s3$ is calculated for all obstacle cells with $s2 > 0$ (which excludes the true obstacles). Figure 4 illustrates the application of the CA rule for 7 time steps, until all the cells with $s1 = Obstacle$ and $s2 > 0$ have the value of $s3$ calculated. Black cells indicate the true obstacles, i.e. the cells identified in the image processing as obstacles. The red, yellow and green ones indicate the enlarged obstacle cells (x equal to 1, 2 and 3, respectively). The robot's initial position is a yellow cell ($x = 2$). In Figure 4.a one can observe $s3$ values after the initialization rule is applied to identify cells at the borders (i.e. obstacle cells with at least one free cell as neighbor). These cells assume $s3 = V$ ($V = 100$ in this example). In Figure 4.b one can observe the value of $s3$ after applying the CA rule once. Figure 4.c shows the final values after NFCD diffusion.

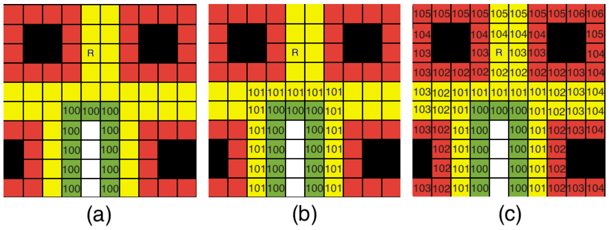


Figure 4: NFCD diffusion: a) initialization $t = 0$, b) $t = 1$ c) Final

Phase 4 - constructs the final path between the robot's current position and the goal. The last phase use all the spread distances to choose a collision free route. The planned route is calculated by either the *DR Procedure* or *EDR Procedure* explained below:

DR Procedure: Constructing just Descending Routes

If the third phase was not necessary, the planned path will be calculated as the original model: any route starting from initial position of the robot until the goal cell, using only free cells, decrementing 1 in the value of $s3$ (GD) at each step of the path. In the case of draw when choosing the next free cell (that is, there are at least two neighbors with a decreasing GD), we implemented some options of conflict decisions: random choice, or chosen the next in a predefined order (clockwise or anticlockwise) or even chosen the next cell that not provokes a robot rotation (that is, it keeps the current robot's orientation). The conflict decision is a parameter of the algorithm.

EDR Procedure: Mixing Escaping and Descending Routes

If the third phase was applied, the planned route is divided in two parts: the first one is obtained using the $s2$ and $s3$ values of obstacle cells (NTOD and NFCD) and the second part is obtained using the $s3$ values of free cells (GD). The initial part of the path must be a route to move the robot from the enlarged obstacles region as fast as possible (or the "escape route"). As such, the algorithm starts from the initial position cell and choose as the next step the neighbor cell with the smallest $s3$ value (i.e. the neighbor with the smallest NFCD). If there is a draw involving neighbors, the algorithm chooses the neighbor with the highest $s2$ (NTOD) value (i.e. between the tied cells it chooses the one more distant from true obstacles). If the draw persists, conflict decisions criteria can be applied, as explained in DR Procedure. The next steps of the route are defined in this way until the algorithm find a cell with $NFCD = V$, which indicates that it is at a border cell in respect to the obstacle regions. The next step of the path will be the neighbor free cell with the smallest GD. Starting from this point, GD stored in $s3$ for free cells will be used to construct the second part of the path, that is, the "descending route".

Figure 5 presents an example of path generated by simulating the new planning algorithm. It shows the complete route formed by the junction of the escape route (in purple) from the cul-de-sac scenario and the descending route (in blue) toward to the goal. Figure 6 shows a summary of the new path planning model algorithm described in this section. It is applied at each n time steps.

One may argue that the escaping route calculus is not necessary, and a simple solution to this problem could be associating high costs (GD) to cells close to obstacles excluding Phase 3 of the algorithm and simplifying Phase 4. Indeed if this approach is used the descending route will be generated as in the new algorithm. The difference in such situation is that the descending route is calculated here in a direct way, just disregarding the closest cells in the calculus. On the other hand, considering critical situations where the robot is positioned within a close distance to the obstacles, the use of the distance to the next free cell for calculating escaping routes makes it a crucial difference. The calculus focuses on keeping the robot away from the enlarged region in the smallest number of steps possible. In that way GD is ignored and an optimal route based on NFCD is found. Besides, several optimal rules (NFCD) are expected since there are several free cells. Thus, the new algorithm uses a second measure (NTOD) to choose between the NFCD optimal routes one with the maximum distance to obstacles. As a result, it finds short escaping routes passing on a safe distance from obstacles.

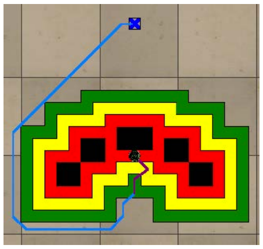


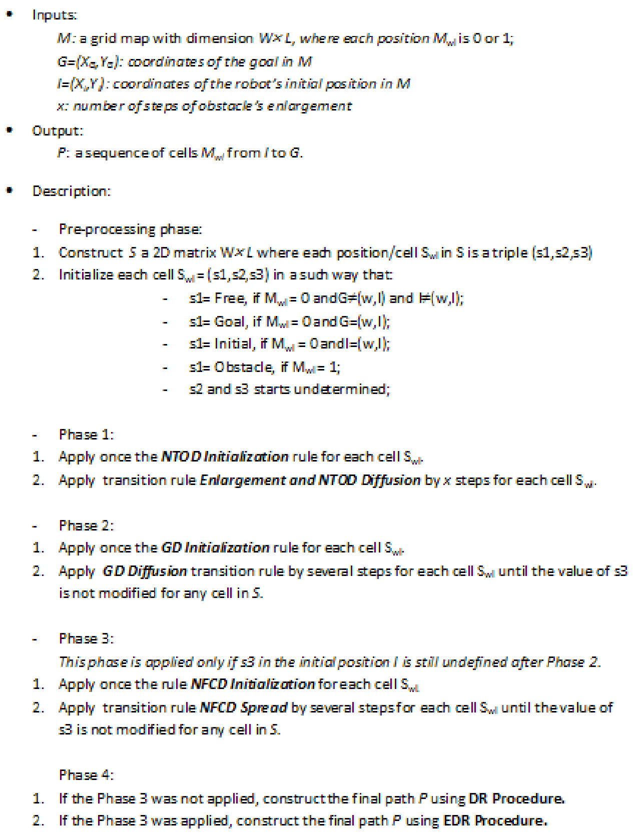
Figure 5: A complete path in a V-REP simulation generated using the new algorithm: escape route (in purple) and descending route (in blue). A blue x represent the goal.

Simulation and Results

A series of simulations were performed aiming at analyzing the behavior of the robot with the proposed adaptive path planning algorithm. Figures 7 shows two simulations using a scenario with 8 obstacles. Figure 7.a illustrates the robot path from (Behring et al., 2000) and Figure 7.b the new path using the adaptive path planning algorithm. A yellow "x" represents the starting point and a blue "x" represents the goal.

In the first simulation (Figure 7.a), the robot final position is very distant from the goal. On the other hand, the e-puck could reach the goal when the new model was used (Figure 7.b). One can see that in the first navigation, the robot has passed closest to obstacles, while in the second one, the robot kept a safe distance from them. In fact, in both simulations the e-puck invaded the enlarged obstacle region ($x = 4$). However, while using the original model the robot continued its route getting even closer to the obstacles.

By using the adaptive planning model, the e-puck robot was able to identify this invasion and to generate an escaping route, correcting its navigation to avoid the obstacles. The green line in Figure 7.b indicates a point in which the re-planning was needed.



- Inputs:

  M: a grid map with dimension *W×L, where each position* M_{wl} is 0 or 1;
  G=(X

- Output:

  P: a sequence of cells M_{wl} from *I* to *G*.

- Description:

  - Pre-processing phase:
  1. Construct *S* a 2D matrix *W×L* where each position/cell S_{wl} in S is a triple (s1,s2,s3)
  2. Initialize each cell S_{wl} = (s1,s2,s3) in a such way that:
     - s1= Free, if M_{wl} = 0 and G≠(w,l) and I≠(w,l);
     - s1= Goal, if M_{wl} = 0 and G=(w,l);
     - s1= Initial, if M_{wl} = 0 and I=(w,l);
     - s1= Obstacle, if M_{wl} = 1;
     - s2 and s3 starts undetermined;

  - Phase 1:
  1. Apply once the ***NTOD Initialization*** rule for each cell S_{wl}.
  2. Apply transition rule ***Enlargement and NTOD Diffusion*** by *x* steps for each cell S_{wl}.

  - Phase 2:
  1. Apply once the ***GD Initialization*** rule for each cell S_{wl}.
  2. Apply ***GD Diffusion*** transition rule by several steps for each cell S_{wl} until the value of s3 is not modified for any cell in *S*.

  - Phase 3:

    This phase is applied only if s3 in the initial position I is still undefined after Phase 2.
  1. Apply once the rule ***NFCD Initialization*** for each cell S_{wl}.
  2. Apply transition rule ***NFCD Spread*** by several steps for each cell S_{wl} until the value of s3 is not modified for any cell in *S*.

    Phase 4:
  1. If the Phase 3 was not applied, construct the final path *P* using **DR Procedure.**
  2. If the Phase 3 was applied, construct the final path *P* using **EDR Procedure.**

Figure 6: Summary of the adaptive path planning model algorithm.

It should be noted that several obstacle collisions were observed during simulations using the original model. Moreover, the robot made several rotations and steps to achieve the goal. On the other hand, using the adaptive planning model the robot performed a smooth navigation, going around the obstacles while keeping a safe distance.

Aiming at better exploring the differences between both models we performed a series of experiments using the scenario presented in Figure 7. Since simulations are non-deterministic, each algorithm was run for 30 simulations in the same scenario and some statistical results were obtained. In comparison, the average of the total distance navigated (in meters) was {8.33;7.82}. The first value represents the original model, whereas the second value represents the new model respectively. The average of the final distance to the goal (in meters) was {0.97;0.03}. The average of the total navigation time (in seconds) was {382;411} and the number of collisions observed during simulations was {7;0}. The final distance indicates that the robot stopped very distant from the goal when the original method is employed. We could observe 7 collisions in 30 runs when using the original model. One can also observe that the adaptive planning did not cause a severe impact on the total time: on average 8% of increase in both scenarios.

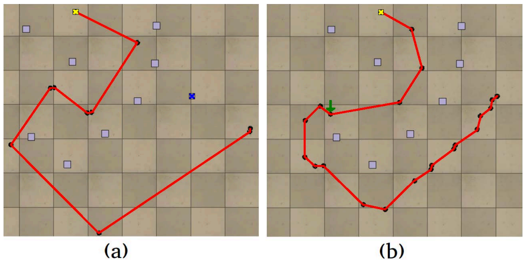

(a)
(b)

Figure 7: (a) Behring's model and (b) Proposed model.

We created another 9 simulation scenarios with a different number of obstacles (Figure 8) and also made a comparative analysis using 30 simulation runs in each scenario. The new adaptive model performance presented: (i) an average reduction of 82% relative to the final distance to the goal (at least 70% of reduction was observed in any scenario); (ii) an average reduction of 5.8% relative to the total navigation distance (in all simulations a decrease was observed); (iii) an average increase of 7% in the total navigation time (in all scenarios the time was increased, but none above 10%) and (iv) no collisions. Therefore, we can concluded based on the results of 300 simulation runs (30 for each scenario) that by applying the new model it was possible to substantially reduce the final distance between the robot and the goal with an additional reduction on the total navigated distance and avoiding obstacles and a very small increase in the total navigation time.

It is important to highlight that the re-planning strategy is used here to correct a control problem since it was possible to observe in our simulations that the robot is not able to navigate as planned. Indeed, one could expect a difference between the planned and the executed route even if more sophisticated control strategies were employed. Using re-planning is a step towards a solution to some dynamic environments where re-planning might be necessary (Wystrach et al., 2014) (Collett et al., 2014).

Discussion

Path planning is a very relevant task for robot autonomous navigation and different techniques have been studied so far (Siegwart et al., 2011). This work investigates an adaptive path planning model based on cellular automata using three different metrics: Goal Distance (GD), Next Free Cell Distance (NFCD) and Next True Obstacle Distance (NTDO). A new plan is constructed at each n steps, starting from the robot's current position, which is extracted from an environment pre-processed image (Behring et al., 2000). The path can be composed by descending and escaping routes whenever is necessary. Descending routes are generated using only GD. Escaping routes are generated using NFCD and

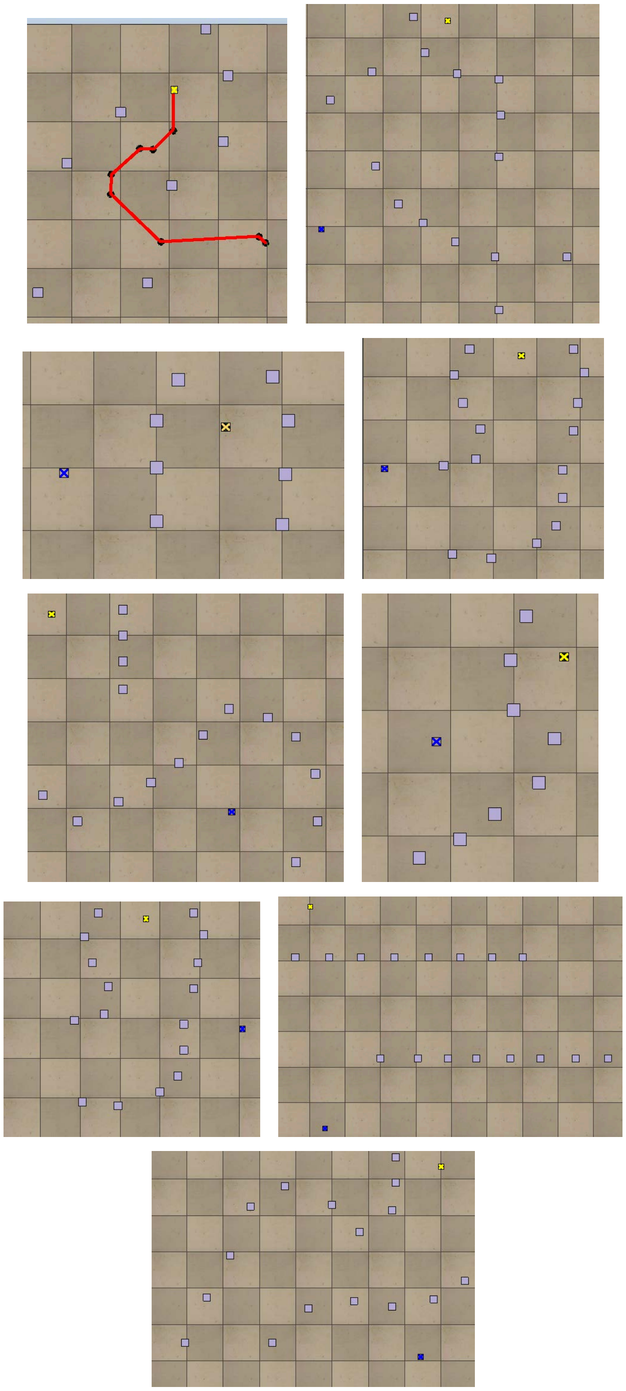


Figure 8: Additional 9 scenarios used in V-REP simulations.

NTOD by avoiding passing too close to the obstacles.

Distance diffusion performed by CA rules returns similar results to the ones obtained by wavefront algorithms in path planning (Zelinsky et al., 1993). An advantage of using CA to perform diffusions is that they are very decentralized structures which enable us to implement them in a parallel way, making them appropriate, for example, to massive parallel architectures as FPGAs (Halbach and Hoffmann, 2004). Moreover, CA performs information spreading in a bottom-up approach instead of a sequential standard wavefront algorithm. This way of modeling enables us in this work to mix different information (GD, NFCD and NTOD metrics) to construct routes with different requisites.

Therefore, GD calculus finds optimal descending paths. Considering escaping routes, the algorithm always finds optimal routes in respect to NFCD and between all possible optimal NFCD routes one with the maximum NTOD. Thus, the escaping route is a point of the Pareto front considering the bi-objective problem (NFCD and NTOD). The algorithm complexity is dominated by GD propagation both in the previous work (Behring et al., 2000) and in the new proposition. Therefore, the full re-planning has the same asymptotic complexity of the previous model. On the other hand, using static environments as mentioned before, at each n steps just phase 4 - which has a simpler complexity than GD propagation (phase 2) - is needed to be performed. This simple complexity analysis justify why we did not observe a significant difference considering the total navigation time comparing the previous model (Behring et al., 2000) and our new proposition.

It is important to point out that full re-planning is not necessary in static environments. Indeed the robot does not do full re-planning, only a new route is calculated given the robot's current position. GD, NFCD and NTOD metrics are not necessary to be recalculated at each n steps, if the environment does not change. One can compute all distances in advance because they are independent of the robot current position.

Conclusion

This current work improves previous research on CA models and path planning problems for a single robot (Behring et al., 2000). In this work we investigated the behaviour of a single robot in a static scenario. Using the V-REP simulator, the current environment configuration (including robot's position), can be easily obtained using the software functionalities. For an effective analysis of the new adaptive model in real robots, it is also necessary to improve the pre-processing image system and investigate the performance on a real robot.

This work is the first step towards merging the proposed model with previous successful models for cooperative robot swarms (Ioannidis et al., 2011) (Ferreira et al., 2014) creating a hybrid CA model that could use local and distributed adaptive path planning tacking similar but more challenging dynamic problems. In static scenarios, few cells change their state at each propagation step. However, the processing performed in each cell is very simple, which turns our algorithm a candidate to be implemented in massive distributed architectures as FPGA that could be incorporated as a dedicated hardware. Our main goal is to apply a CA that would resemble the desert ants behaviour where re-planning is a necessary condition for survival in a dynamic environment (Wystrach et al., 2014) (Collett et al., 2014).

Future work includes: (1) merging this new model with already established models for cooperative robots (Ioannidis et al., 2011) (Ferreira et al., 2014) plus the use of real e-puck robots; (2) using evolutionary robotics techniques (Vargas et al., 2014) to adjust the parameters involved in the model, as the value of n (number of steps to re-planning) and others related to the implementation of the robot's dynamics, and finally (3) adapting the new model to planning paths in environments with mobile obstacles. In fact, our adaptive path-planning model is a step toward dealing with dynamic scenarios and multiple robots.

Acknowledgements

GMBO is grateful to CNPq, CAPES and Fapemig support.

References

Akbarimajd, A. and Hassanzadeh, A. (2012). Autonomously Implemented Versatile Path Planning for Mobile Robots Based on Cellular Automata and Ant Colony. *International Journal of Computational Intelligence Systems*, 5(1):39–52.

Arkin, R. C. (1998). *Behavior-based robotics*. MIT press.

Behring, C., Bracho, M., Castro, M., and Moreno, J. A. (2000). An Algorithm for Robot Path Planning with Cellular Automata. In *Proceedings of the Fourth International Conference on Cellular Automata for Research and Industry: Theoretical and Practical Issues on Cellular Automata*, pages 11 – 19.

Collett, T. S., Lent, D. D., and Graham, P. (2014). Scene perception and the visual control of travel direction in navigating wood ants. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 369(1636):20130035.

EPFL (2015). E-puck education robot. http://www.e-puck.org.

Ferreira, G. B., Vargas, P. A., and Oliveira, G. M. (2014). An improved cellular automata-based model for robot path-planning. In *Advances in Autonomous Robotics Systems*, pages 25–36. Springer.

Halbach, M. and Hoffmann, R. (2004). Implementing cellular automata in fpga logic. In *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, page 258. IEEE.

Hernández, C., Sun, X., Koenig, S., and Meseguer, P. (2011). Tree adaptive a*. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 123–130. International Foundation for Autonomous Agents and Multiagent Systems.

Ioannidis, K., Sirakoulis, G. C., and Andreadis, I. (2011). A Path Planning Method Based on Cellular Automata for Cooperative Robots. *Applied Artificial Intelligence*, 25(8):721–745.

Kostavelis, I., Boukas, E., Nalpantidis, L., and Gasteratos, A. (2012). Path Tracing on Polar Depth Maps for Robot Navigation. *Cellular Automata*, pages 395–404.

Marchese, F. M. (2002). A directional diffusion algorithm on cellular automata for robot path-planning. *Future Generation Computer Systems*, 18(7):983–994.

Oliveira, G. M., Martins, L. G., Ferreira, G. B., and Alt, L. S. (2010). Secret key specification for a variable-length cryptographic cellular automata model. In *Parallel Problem Solving from Nature, PPSN XI*, pages 381–390. Springer.

Parker, L. E., Birch, B., and Reardon, C. (2003). Indoor target intercept using an acoustic sensor network and dual wavefront path planning. In *In Proceedings of IEEE International Symposium on Intelligent Robots and Systems (IROS 03*, pages 278–283.

Phillips, M., Likhachev, M., and Koenig, S. (2014). Pa* se: Parallel a* for slow expansions. In *Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling*, pages 208–216.

Robotics, C. (2015). V-rep, virtual robot experimentation platform. http://www.coppeliarobotics.com.

Sarkar, P. (2000). A brief history of cellular automata. *ACM Computing Surveys (CSUR)*, 32(1):80–107.

Shu, C. and Buxton, H. (1995). Parallel path planning on the distributed array processor. *Parallel Computing*, 21(11):1749–1767.

Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D. (2011). *Introduction to autonomous mobile robots*. MIT press.

Swiecicka, A., Seredynski, F., and Zomaya, A. Y. (2006). Multiprocessor scheduling and rescheduling with use of cellular automata and artificial immune system support. *Parallel and Distributed Systems, IEEE Transactions on*, 17(3):253–262.

Tavakoli, Y., Javadi, H. H. S., and Adabi, S. (2008). A cellular automata based algorithm for path planning in multi-agent systems with a common goal. *International Journal of Computer Science and Network Security. v8*, pages 119–123.

Tinbergen, N. (1932). Ueber die orientierung des bienenwolfes (philanthus triangulum fabr.). *Journal of Comparative Physiology A: Neuroethology, Sensory, Neural, and Behavioral Physiology*, 16(2):305–334.

Vargas, P. A., Di Paolo, E. A., Harvey, I., and Husbands, P. (2014). *The horizons of evolutionary robotics*. MIT Press.

Wystrach, A., Philippides, A., Aurejac, A., Cheng, K., and Graham, P. (2014). Visual scanning behaviours and their role in the navigation of the australian desert ant melophorus bagoti. *Journal of Comparative Physiology A*, 200(7):615–626.

Zelinsky, A., Jarvis, R., Byrne, J., and Yuta, S. (1993). Planning paths of complete coverage of an un- structured environment by a mobile robot. In *Proceedings of International Conference on Advanced Robotics*, pages 533–538.

The Evolution of Assortment with Multiple Simultaneous Games

S. Tudge[1], A. Jackson[1], R. Watson[1] and M. Brede[1]

[1] The University of Southampton, Southampton SO17 1BJ, UK
sjt4g11@soton.ac.uk

Abstract

Current theories of social evolution predict the direction of selection for a given level of assortment. What remains unclear is how to determine the direction of selection on assortment itself if this were subject to evolutionary change. Here we define and analyse a simple model that allows us to investigate the evolution of assortment. We find that there is only a positive selection gradient for increased assortment if the population is polymorphic in the cooperative trait. We further show that if the individuals in question engage in multiple cooperative dilemmas simultaneously then there may be a continued selection on increased assortment which is ultimately sufficient to resolve severe dilemmas such as the prisoner's dilemma.

Introduction

The evolution of cooperation was a problem which Darwin labelled "his one special difficulty". In a naïve interpretation the presence of cooperation (and particularly altruism) presents a fundamental challenge to the Darwinian view of nature. Why would individuals be selected to perform actions which are beneficial to others at a cost to themselves? The two major attempts at answers to this question come in the form of kin selection (Hamilton, 1964; Gardner et al., 2011; Queller, 1985; Maynard Smith, 1964) and group selection (Wilson, 1975; Borrello, 2005). These two processes have been shown to be mathematically equivalent (Lehmann et al., 2007b; Foster, 2006; Queller, 1992) as they both essentially depend on population structure that gives rise to assortment of interactions. Here assortment means that like individuals will interact more often than would be expected from random interactions. Self-interested individuals will cooperate in an assorted population because, by virtue of being a cooperator, they are more likely to receive the benefits of other cooperators. In agreement with a number of authors we see assortment as the key factor in the evolution of cooperation (Eshel and Cavalli-Sforza, 1983; Fletcher and Zwick, 2006; Godfrey-Smith, 2008; Michod and Sanderson, 1985; Sober, 1992).

An outstanding puzzle for the theoretician studying the evolution of cooperation is to investigate the evolution of assortment/relatedness. In many instances in nature individuals may have traits which in effect modify which other members of the population they interact with. For instance, dispersal rate (Smaldino and Schank, 2012; Pepper and Smuts, 2002; Lehmann et al., 2007a) may have a genetic component and hence be subject to selection. The vast majority of studies of the evolution of cooperation take such parameters as given; a more complete understanding of the evolution of cooperation would be facilitated by studying, in the most general and therefore most abstract setting possible, how such genetic traits effecting assortment will coevolve with genetic traits determining social behaviours such as cooperation. A small number of recent papers have begun to look at such processes. Notably Powers et al. (2011) study a model in which individuals play a public goods game in a group structured population. In addition to a gene controlling the social strategy (i.e. cooperate or defect) they also look at the concurrent evolution of a gene which determines a group size preference of the individuals. Individuals disperse and join new groups, the sizes of these groups are determined genetically, thus individuals may prefer to be in either larger or smaller groups. Because a population composed of small groups is more highly assorted than a population composed of large groups; this "group size preference" has the effect of an assortment parameter. In such a model it is found that the coevolution of population structure and social strategy leads to a feedback process whereby the evolution of cooperation and the evolution of social structures that support cooperation facilitate each other so that large levels of cooperation are eventually selected for.

Jackson and Watson (2015) use the formalism of a metagame to investigate the evolutionary dynamics of game changing behaviour such as assortment. In their model each agent has a genetically determined payoff matrix representing a desired game, as well as a gene determining their social strategy. Unlike conventional game theoretic studies, which simply investigate the dynamics of a given game, this model allows for the underlying game to be altered via the process of natural selection. They find that a strong linkage disequilibrium emerges, whereby cooperators choose a game that is favourable to cooperators and likewise for defectors. Which of these two strategies ultimately wins depends upon the equilibrium properties of the game. If the equilibrium of the initial game is dominated by cooperators then selection will change the game towards the harmony game, thereby further entrenching cooperation. However, if defection dominates at equilibrium then selection will move the game towards a more severe dilemma, and thus further entrench defection. For selection to have any effect on the underlying game there must be some polymorphism in the social strategy, which is not the case in the prisoner's dilemma at equilibrium.

One of the intermediate aims of this paper is to present a model for the evolution of assortment which is more abstract, and hence more general, than any previous model. This is desirable as it allows one to dispose of many arbitrary modelling assumptions. We look at the coevolution of assortment and social strategy in a cooperative dilemma and investigate under what conditions there is a positive selection gradient on increased assortment. In agreement with previous studies we find that such a gradient only exists if there is currently a polymorphic level of cooperation in the population. In the language of two-player games this is a situation represented by a snowdrift game. We find that games, such as the prisoner's dilemma, which have no cooperation at equilibrium, do not result in a selection for increased assortment. Thus, such a mechanism cannot "resolve" a prisoner's dilemma. Note that the prisoner's dilemma game represents the biological scenario of strong altruism, which is often observed in nature (West et al., 2007).

The second contribution of this paper is to show a plausible scenario in which assortment can be increased sufficiently by selection to levels high enough to "solve" the prisoner's dilemma (or in other words levels high enough to observe strong altruism). The principal idea is that individuals will be engaged in multiple social interactions at once, assumed to be controlled by genes at different loci. Thus, any two individuals will engage in a large number of social interactions simultaneously. A simple example may be a species of bacteria that may produce a number of public goods, each of which is simply a protein. Each individual bacterium may or may not produce each public good. Thus, the multiple interactions within the species can be represented via a series of games (rather than conventional studies which consider only one game taking place). Each individual may be a cooperator or a defector in each game independently of whether or not they cooperate or defect in other games. In such an instance it may be the case that one of these games is a snowdrift game and hence provides a positive selection gradient for increased relatedness until a sufficient level of assortment has arisen to fixate cooperation at this locus. As a bi-product of this process other games will become transformed such that they are then polymorphic (i.e. contain a mixture of both cooperators and defectors). Given enough games between individuals there will be a continual selection on increased assortment so that the population ends up highly assorted and therefore cooperation can evolve even for much more severe dilemmas.

Very few authors have looked at the possibility of the outcome of multiple games being played at once or sequentially between agents/individuals. Those that do, do so from within economics or psychology. In particular a number of authors (Bednar and Page, 2007; Bednar et al., 2012; Grimm and Mengel, 2012) have looked at the consequences of multiple, and qualitatively different, games being played in sequence between subjects. The key themes of these papers tend to be to do with cognitive spill-over, i.e. how the outcome of one game might affect another. Otherwise they are to do with the cognitive load on the individual i.e. how individuals might use heuristics or rules learnt in one game to reduce the computation needed to solve other games. To the best of our knowledge no authors have looked at the dynamics of multiple games from within evolutionary game theory. One reason for the lack of such a study is that the basic result, i.e. that each game reaches the ESS independently, is not interesting unless one allows for some manner of epistatic interaction between games. In our model the epistasis comes via the intermediary of the evolving assortment parameter. With this interdependence the presence of multiple games results in a qualitatively different outcome, as we shall show.

Model

We shall restrict our analysis to those social interactions which can be represented via pair-wise interactions. Such interactions can be represented via a two-player game. Santos et al. (2006) show that the space of all possible two-player, two-strategy, symmetric games is two dimensional. The payoff matrix for which is given by:

$$M = \begin{pmatrix} 1 & S \\ T & 0 \end{pmatrix} \tag{1}$$

In which the reward for mutual cooperation is normalised to one, and the punishment for defection to zero. T parameterises the temptation to defect against a cooperator, and S the sucker's payoff received by cooperating against a defector.

One can neatly handle the effects of assortment via a transformation of the game. A level of assortment, α, is defined as follows: with probability α an individual is paired with another individual with the same strategy as itself, and with probability $1-\alpha$ it is paired with a random individual. It can be shown (van Veelen, 2011) that the outcome of a game M under assortment α is equivalent to the outcome of the game M' with no assortment, where:

$$M'_{ij} = \alpha M_{ii} + (1-\alpha) M_{ij} \tag{2}$$

Thus one can consider assortment as an effective transformation of the game into a more harmonious one, as figure 1

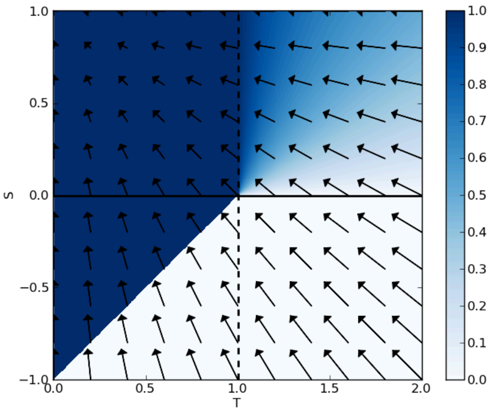


Figure 1: The space of all two-player symmetric cooperative dilemmas and the effects of assortment. Each point in the space represents a two player game parameterised via S and T, the colour represents the equilibrium level of cooperation reached from an initial condition of one half cooperators (one represents cooperate and zero defect). Each arrow represents the effective transformation of the game under assortment of $\alpha = 0.07$. Assortment has the effect of transforming the game towards the Harmony (top left) region of game space, thus making it more cooperative.

illustrates.

Agents play many games simultaneously, the strategy in each is determined at a separate locus. It is thus possible for an agent to cooperate in some games, whilst defecting in others. All agents play all games with all other agents. The overall payoff they receive is simply the mean of the payoffs from all games.

There are N_G games being played at once. An array of matrices determines all of the games such that the k*th* game is given by:

$$M_k = \begin{pmatrix} 1 & S_k \\ T_k & 0 \end{pmatrix} \tag{3}$$

Each value of S and T is chosen at random from the uniform distribution $S \in [-1,1]$ and $T \in [0,2]$.

In addition to social strategies in multiple games the agents have a gene for a desired level of assortment, α_i. With probability α_i agent i interacts with a clonally related individual, that is it has the same value of the gene at all loci. With probability $1-\alpha_i$ it enters a pool of players, and therefore interacts with an agent chosen randomly from the subset of those other individuals who have also chosen to enter the pool.

Selection proceeds in a generational GA using fitness proportionate selection. Mutation may occur at a locus controlling social strategy; with probability μ_s each locus changes to its opposite strategy. The assortment gene is mutated with probability μ_a, and subsequently changes by an amount drawn from the normal distribution with mean zero and standard deviation 0.05 (the results do not depend sensitively on the particular choice of these parameters). If the value mutates outside of the permitted range it is simply scaled back to zero/one. We assume that the time scale of evolution is slower for assortment than it is for game strategy, so that the game strategy is always in equilibrium and that mutations in α occur gradually. We also assume that the primitive state of the population is freely mixed. We thus begin our simulation with $\alpha = 0$ for all agents, and allow strategy frequencies to reach equilibrium before ‘turning on’ a small mutation in α.

In addition to this there is also a small cost to being assorted, $k \times \alpha_i$, which increases linearly with the agents assortment (typically $k = 1 \times 10^{-3}$ unless otherwise stated). This cost is introduced to ensure that all change in assortment is adaptive, rather than being due to drift, when all else is equal then selection will not favour an increase in assortment. This cost is kept small so as not to effect the direction of selection significantly in cases other than drift.

Results

We proceed by investigating special cases of the more general model. Firstly we look at a model in which assortment evolves but there is only one game being played. Secondly, we look at multiple games, but in the absence of assortment. Finally, we analyse the full model with both evolvable assortment and multiple games.

The Evolution of Assortment with a Single Game

We first sketch a mathematical argument, and go on to compare it with a simulation based model.

Let the frequency of cooperators in the pool be p_C and likewise for defectors p_D. Then the payoff an individual with strategy i gets (as a function of α) is:

$$\pi_i(\alpha) = \alpha M_{ii} + (1-\alpha)\left[p_C M_{iC} + p_D M_{iD}\right] \tag{4}$$

p_C is calculated by taking a mean of the number of cooperators, weighted by the chance that they enter the pool, which is $1-\alpha_i$. E_C is the expected *number* of cooperators in the pool:

$$E_C = \sum_i s_i(1-\alpha_i) \tag{5}$$

Where $s_i = 1$ for a cooperator and 0 for a defector. The sum runs over all members of the population. Similarly for defectors:

$$E_D = \sum_i (1-s_i)(1-\alpha_i) \tag{6}$$

It follows that:

$$p_C = \frac{E_C}{E_D + E_C} \tag{7}$$

and

$$p_D = \frac{E_D}{E_D + E_C} \tag{8}$$

To investigate this model further one simply needs to determine when individuals with a slightly larger than normal level of desired assortment can invade a population. We consider a population composed of individuals who all have the same value for desired assortment α, and then simply asked whether a mutant with a slightly larger value of desired assortment $\alpha + \delta\alpha$ can invade this population. This can be more simply answered by looking at payoff and asking whether or not this is an increasing function of α. The payoff to cooperators is given by:

$$\begin{aligned} \pi_c(\alpha) &= \alpha + (1-\alpha)\left[p_C + Sp_D\right] && (9) \\ &= (1 - p_C - Sp_D)\,\alpha + p_C + Sp_D && (10) \end{aligned}$$

Because $S < 1$ it follows that $(1 - p_C - Sp_D) > 0$ and thus $\pi_i(\alpha)$ is an increasing function of α, which means that there is always a selection pressure for existing cooperators to increase their values of α_i.

What about if the population consists entirely of unassorted defectors? We then wish to know the value of α for which cooperators can invade, i.e.:

$$\begin{aligned} \pi_c(\alpha) &> \pi_d(0) && (11) \\ \alpha + (1-\alpha)S &> 0 && (12) \end{aligned}$$

because we consider cooperators invading in infinitesimal quantities we assume $p_C = 0$ and also $p_D = 1$. Equation (12) reduces to:

$$\alpha > \frac{S}{S-1} \qquad (13)$$

Therefore, in the limit of infinitesimal increase in α, cooperators can only invade if $S > 0$ (found by setting $\alpha = 0$ in the above equation); that is α will only increase in a snowdrift game.

The above argument is a sketch of a formal mathematical argument in which a number of simplifying assumption were necessary. We complement this argument with a agent based genetic algorithm, which includes stochastic effects not present in the mathematical model. We run the full simulation model, but set $N_G = 1$. We record the level of mean α and mean cooperation at equilibrium and plot these over the space of all possible games on the TS plane. The results are presented in figure 2.

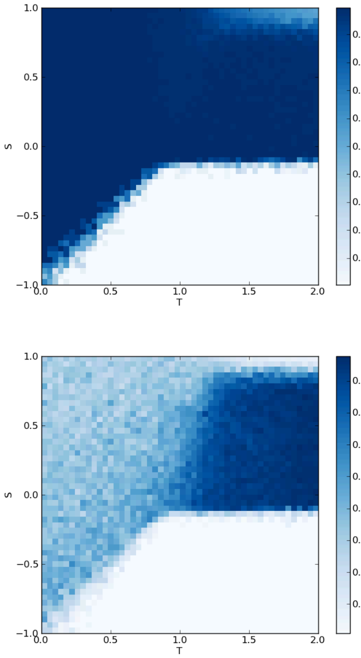


Figure 2: Equilibrium levels of cooperation (top) and α (bottom) over the space of all possible games. Only snowdrift games provide a positive selective gradient on α.

The conclusion of the model is that a positive selection for assortment will only occur if there exists some preliminary level of cooperation at equilibrium. As there is mutation on strategy there is also a small amount of selection pressure to increase assortment even in games which are totally dominated by cooperators (as is the case in the harmony game and some of the stag hunt game). This is to protect against the occasional introduction of defectors into the population. In the snowdrift regions in which the population is almost all cooperate, and also in the region in which $S \simeq 1$ there is little increase in assortment. Although a purely mathematical argument asserts that there will be selection for assortment, stochastic effects and the small cost to assortment counteract this effect in the marginal cases.

Multiple Games in the Absence of Evolvable Assortment

This model is simply the full model minus the possibility of evolvable assortment. Thus, each allele represents the social strategy in an independent game. We find that the equilibrium level for each game is simply equal to the game's ESS. In other words the presence of multiple games does not affect the outcome of selection. This is to be expected as there is no epistasis between the different alleles. Figure 3 shows a typical output of this model. For ease of interpretation we create a scatter plot of the games being played on the TS plane alongside the evolution of strategy frequencies.

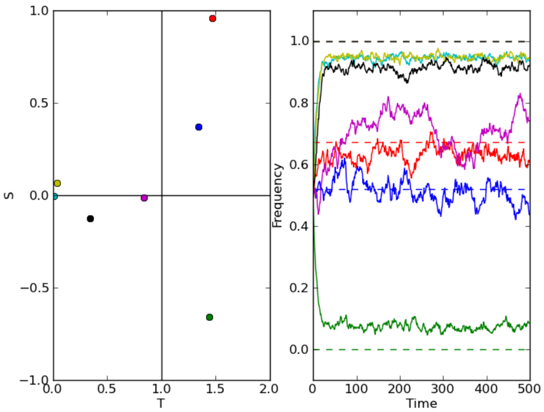


Figure 3: Many random games being played. We record the frequency of the population that plays cooperate at each allele. This matches predictions for the ESS of each game independently (dotted lines).

The Evolution of Assortment with Multiple Games

Finally we present findings for the full version of the model, in which assortment is subject to selection, and the agents engage in multiple games. We choose seven random points on the TS plane and let the simulation run to equilibrium. The key finding here is that the presence of multiple games allows for a continuous selective pressure towards increased assortment. Each social strategy gene is not directly affected by other social strategy genes. However, there is epistasis between them, but via the intermediary of the assortment allele. Any game that is polymorphic (i.e. is a snowdrift game) creates a selection pressure on increased assortment. This occurs until the game is transformed into a Harmony game. Other, non-polymorphic games do not give rise to any selection pressure on assortment. However, a certain level of assortment may transform a prisoner's dilemma into a snowdrift game, thus engaging further selection pressure for increased assortment. This process may repeat until a level of assortment has evolved sufficient to resolve all such dilemmas. Whether this occurs depends upon the details of where the various games lie, but the probability of at least one game being polymorphic clearly increases with the number of games being played. Figure 4 shows one example of how this process may work, and one example of when it fails to do so. In the example in which there is no significant increase in assortment the initial distribution of games does not include a snowdrift game, whereas in the other example the initial distribution does include a snowdrift game, thus eventually providing enough assortment to resolve the prisoner's dilemma. This particular instance of the model uses a population size of 1028 and $\mu_a = \mu_T = 5 \times 10^{-3}$ and is run for 1600 generations.

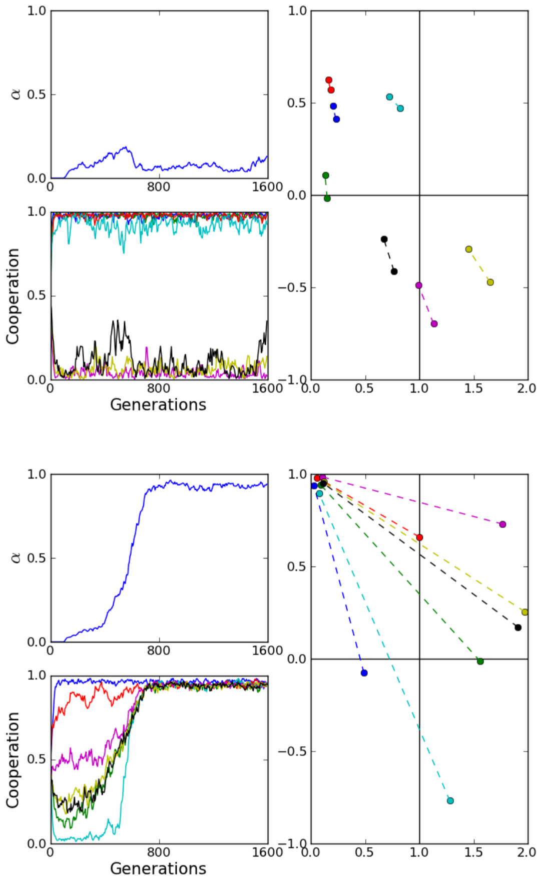


Figure 4: The dynamics of a simulation with 7 randomly chosen games. In each panel: top left: change in the mean value of α over time. Bottom left: change in allele frequency for each game locus. Right: position of games in ST space, dotted lines represent the effective transformation of the game due to increasing α.

Discussion

It is known in the field of evolutionary biology that population structure is important when considering the evolution of social traits (Eshel and Cavalli-Sforza, 1983). Specifically positive assortment can allow for the evolution of cooperative or altruistic behaviour. Formalisms such as kin selection (Hamilton, 1964; Grafen, 1982; Maynard Smith, 1964) and multi-level selection (Price, 1970; Okasha, 2009) allow one to make precise calculations of the expected change in the frequency of a cooperative allele due to selection (see for example Gardner et al. (2011)). Furthermore, the natural world is full of examples of cooperative behaviour, such as the cooperation between cellular slime moulds (Strassmann et al., 2011), eusocial insects (Hölldobler and Wilson, 1990) or the cells of a multicellular organism (Michod and Roze, 2001; Buss, 1987; Queller, 2000). In many cases the population structure of such organisms may have a genetic component and thus be subject to evolutionary change. For example, trees may alter the dispersal radius of their seeds (Pepper and Smuts, 2002), birds may alter the time at which they leave their natal group (Bulmer, 1994), social wasps may alter the number of eggs they lay in one host (Ode and Strand, 1995). In addition, multicellular organisms undergo a unicellular bottleneck which may be an adaptation that increases the genetic homogeneity of the organism (Ryan and Watson, 2015). Whilst it is at least plausible that many of these feature are adaptations there lacks a unified theoretical understanding of the conditions under which an increases positive assortment will evolve.

Recently a number of authors (Powers et al., 2011; Jackson and Watson, 2015) have begun to address this issue. One point that has emerged from these studies, and is backed by a formal mathematical argument here, is that selection will not increase positive assortment unless the underlying game is polymorphic. Games such as the prisoner's dilemma are not polymorphic at equilibrium, and therefore selection on assortment cannot "get started". Furthermore, these games represent the biologically prevalent case of strong altruism (see for example Doncaster et al. (2013)). We have presented a potential resolution to this problem: if individuals interact in many social dilemmas simultaneously then there may exist a feedback process whereby the weaker dilemmas transform the stronger dilemmas into weaker ones and eventually all dilemmas are resolved.

Biologically this could represent, for example, the evolutionary path towards multicellularity (see also Michod and Roze (2001); Maynard Smith and Szathmary (1997); Ispolatov et al. (2012); Grosberg and Strathmann (2007); Jablonka and Lamb (2006)). The cells of a proto-organism will eventually need to cooperate in many different manners, such as by producing different public goods with differing costs or by refraining from different forms of selfish reproduction, each of which with a different benefit. Our model shows how this can be thought of as a continuous process, rather than a binary one (see Godfrey-Smith (2009)). The likelihood of this happening is greatly increased as the number of dilemmas being played increases. It seems plausible that the case of individuals playing one single social dilemma is an idealisation, and that in reality individuals, having many genes, and many potential social interactions will usually be engaged in a very large number of social interactions at once, thus making the transition to social living possible.

Acknowledgments

This work was supported by an EPSRC Doctoral Training Centre grant (EP/G03690X/1).

References

Bednar, J., Chen, Y., Liu, T. X., and Page, S. (2012). Behavioral spillovers and cognitive load in multiple games: An experimental study. *Games and Economic Behavior*, 74(1):12–31.

Bednar, J. and Page, S. (2007). Can game(s) theory explain culture? the emergence of cultural behavior within multiple games. *Rationality and Society*, 19(1):65–97.

Borrello, M. E. (2005). The rise, fall and resurrection of group selection. *Endeavour*, 29(1):43–7.

Bulmer, M. (1994). *Theoretical evolutionary ecology*. Sinauer Associates Sunderland, MA.

Buss, L. (1987). *The Evolution of Individuality*. Princeton paperbacks. Princeton University Press.

Doncaster, C. P., Jackson, A., and Watson, R. A. (2013). Manipulated into giving: when parasitism drives apparent or incidental altruism. *Proceedings of the Royal Society B: Biological Sciences*, 280(1758):20130108.

Eshel, I. and Cavalli-Sforza, L. L. (1983). Assortment of encounters and evolution of cooperativeness. *Proceedings of the National Academy of Sciences*, 79(4):1331–1335.

Fletcher, J. A. and Zwick, M. (2006). Unifying the Theories of Inclusive Fitness and Reciprocal Alturism. *The American naturalist*, 168(2):252–262.

Foster, K. (2006). Balancing synthesis with pluralism in sociobiology. *Journal of evolutionary biology*, 19(5):1394–1396.

Gardner, A., West, S. A., and Wild, G. (2011). The genetical theory of kin selection. *Journal of evolutionary biology*, 24(5):1020–1043.

Godfrey-Smith, P. (2008). Varieties of Population Structure and the Levels of Selection. *The British Journal for the Philosophy of Science*, 59(1):25–50.

Godfrey-Smith, P. (2009). *Darwinian Populations and Natural Selection*. OUP Oxford.

Grafen, A. (1982). How not to measure inclusive fitness. *Nature*, 298(5873):425.

Grimm, V. and Mengel, F. (2012). An experiment on learning in a multiple games environment. *Journal of Economic Theory*, 147(6):2220–2259.

Grosberg, R. K. and Strathmann, R. R. (2007). The evolution of multicellularity: A minor major transition? *Annual Review of Ecology, Evolution, and Systematics*, 38(1):621–654.

Hamilton, W. (1964). The genetical evolution of social behaviour. I. *Journal of theoretical biology*, 7(1):17–52.

Hölldobler, B. and Wilson, E. O. (1990). *The ants*. Harvard University Press.

Ispolatov, I., Ackermann, M., and Doebeli, M. (2012). Division of labour and the evolution of multicellularity. *Proceedings of the Royal Society B: Biological Sciences*, 279(1734):1768–76.

Jablonka, E. and Lamb, M. J. (2006). The evolution of information in the major transitions. *Journal of theoretical biology*, 239(2):236–46.

Jackson, A. and Watson, R. (2015). Metagames: A formal framework for the evolution of game-changing behaviours. *In Submission*.

Lehmann, L., Keller, L., and Sumpter, D. J. (2007a). The evolution of helping and harming on graphs: the return of the inclusive fitness effect. *Journal of evolutionary biology*, 20(6):2284–2295.

Lehmann, L., Keller, L., West, S., and Roze, D. (2007b). Group selection and kin selection: two concepts but one process. *Proceedings of the National Academy of Sciences*, 104(16):6736–6739.

Maynard Smith, J. (1964). Group selection and kin selection. *Nature*, 201:1145–1147.

Maynard Smith, J. and Szathmary, E. (1997). *The Major Transitions in Evolution*. OUP Oxford.

Michod, R. E. and Roze, D. (2001). Cooperation and conflict in the evolution of multicellularity. *Heredity*, 86(Pt 1):1–7.

Michod, R. E. and Sanderson, M. J. (1985). Behavioral structure and the evolution of cooperation. *Evolution-Essays in honor of John Maynard Smith*, pages 95–104.

Ode, P. J. and Strand, M. R. (1995). Progeny and sex allocation decisions of the polyembryonic wasp copidosoma floridanum. *Journal of Animal Ecology*, pages 213–224.

Okasha, S. (2009). *Evolution and the Levels of Selection*. Clarendon Press.

Pepper, J. W. and Smuts, B. B. (2002). A mechanism for the evolution of altruism among nonkin: positive assortment through environmental feedback. *The American Naturalist*, 160(2):205–213.

Powers, S. T., Penn, A. S., and Watson, R. a. (2011). The concurrent evolution of cooperation and the population structures that support it. *Evolution; international journal of organic evolution*, 65(6):1527–43.

Price, G. R. (1970). Selection and covariance. *Nature*, 227:520–21.

Queller, D. C. (1985). Kinship, reciprocity and synergism in the evolution of social behaviour. *Nature*, 318.

Queller, D. C. (1992). Quantitative genetics, inclusive fitness, and group selection. *American Naturalist*, pages 540–558.

Queller, D. C. (2000). Relatedness and the fraternal major transitions. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences*, 355(1403):1647–55.

Ryan, P. and Watson, R. A. (2015). The evolution of a unicellular bottleneck in the life history of multicellular organisms. Unpublished.

Santos, F. C., Pacheco, J. M., and Lenaerts, T. (2006). Evolutionary dynamics of social dilemmas in structured heterogeneous populations. *Proceedings of the National Academy of Sciences of the United States of America*, 103(9):3490–4.

Smaldino, P. E. and Schank, J. C. (2012). Movement patterns, social dynamics, and the evolution of cooperation. *Theoretical population biology*, 82(1):48–58.

Sober, E. (1992). The Evolution of Altruism: Correlation, Cost, and Benefit. *Biology and Philosophy*, 7(2):177–187.

Strassmann, J. E., Gilbert, O. M., and Queller, D. C. (2011). Kin discrimination and cooperation in microbes. *Annual review of microbiology*, 65:349–67.

van Veelen, M. (2011). The replicator dynamics with n players and population structure. *Journal of theoretical biology*, 276(1):78–85.

West, S. A., Griffin, A. S., and Gardner, A. (2007). Social semantics: altruism, cooperation, mutualism, strong reciprocity and group selection. *Journal of evolutionary biology*, 20(2):415–432.

Wilson, D. S. (1975). A theory of group selection. *Proceedings of the National Academy of Sciences of the United States of America*, 72(1):143–6.

Exploring the organisation of complex systems through the dynamical interactions among their relevant subsets

Alessandro Filisetti[1,2], Marco Villani[4], Andrea Roli[5], Marco Fiorucci[3] and Roberto Serra[4]
[1]European Centre for Living Technology, Venice
[2]Explora s.r.l, Rome, Italy
[3]Department of Environmental Sciences (DAIS), Università Ca' Foscari
[4]Department of Physics, Informatics and Mathematics, Università di Modena e Reggio Emilia
[5]Department of Computer Science and Engineering (DISI), Università di Bologna
a.filisetti@explora-biotech.com

Abstract

Complex systems often show forms of organisation where a clear-cut hierarchy of levels with a well-defined direction of information flow cannot be found. In this paper we propose an information-theoretic method aimed at identifying the dynamically relevant parts of a system along with their relationships, interpreting in such a way the system's dynamical organisation. The analysis is quite general and can be applied to many dynamical systems. We show here its application to two relevant biological examples, the case of mammalian cell cycle network and of Mitogen Activated Protein Kinase (MAPK) cascade. The result of our analysis shows that the elements of the mammalian cell cycle network act as a single compact group, whereas the MAPK system can be decomposed into two dynamically distinct parts, with asymmetric information flows.

Introduction

Complex systems often show forms of organisation that cannot be understood by referring to simple hierarchical models (like e.g. a tree). In most interesting cases one is faced with complicated situations, sometimes referred to as "tangled hierarchies", where a clear-cut hierarchy of levels, with a unique well-defined direction of information flow cannot be found.

In previous work (Villani et al., 2013; Villani and Serra, 2013; Villani et al., 2015) we have introduced the *Dynamical Cluster Index method* (shortly DCI[1]), which makes use of a measure based on information theory that appears well suited to explore the organisation of such complex systems. The DCI makes it possible to identify Candidate Relevant Subsets of variables (CRS) that show an integrated behaviour, while interacting more weakly with the rest of the system. The DCI has been applied with interesting results to several systems: some of them were artificially designed in order to test the effectiveness of the technique, while others refer to interesting chemical or biological systems (Filisetti et al., 2011; Sarma and Ghosh, 2012; Chaos et al., 2006; Farmer and Kauffman, 1986). Moreover, ongoing and yet unpublished work is also showing promising results in the study of social systems such as the detection of dynamical communities in regional innovation projects studied within the "Emergence by Design"[2] European project, the Tuscany innovation system and the UK consumer price indices.

In many cases, though, the identified CRS have an intricate nested structure, so that it might not be clear which groups of variables are really important. To overcome this problem, a filtering algorithm is needed, which should be able to select those CRS responsible for the dynamics of the system. To this aim, in this paper we introduce a sieving procedure able to extract, from the set of candidate CRS, a smaller number of disjoint or partially overlapping sets, that capture the basic features of the groups of integrated variables, thus reducing the need for user intervention in the interpretation of the outcomes of the method.

In this work we also introduce a further improvement by considering the flow of information among the identified subsets of variables (i.e. relevant subsets RS) that is estimated on the basis of transfer entropy (Schreiber, 2000). In this way it is possible to determine whether there is an information flow from set A to set B, and vice-versa. Moreover, also the direction of the net information flow is defined. We use this method to analyse some artificially designed cases and also two biological models.

The paper is structured as in the following. In the subsequent sections we summarise the theoretical background and describe the sieving procedure and the technique used to detect the information flow among RS. Then, several case studies are presented, while in the final section conclusions and further developments are discussed.

The Dynamical Cluster Index method

In this section we summarise the main features of the DCI, as presented in (Villani et al., 2013, 2015). The DCI is an extension of the Functional Cluster Index (CI) introduced by Edelman and Tononi in 1998 (Tononi et al., 1998) and aimed at detecting functional clusters in brain regions. In our work, we relaxed the stationary constraint and extended the CI to

[1] When clear from the context, the same acronym 'DCI' will be used either to denote the index or the method using it.

[2] EU grant agreement n. 284625

actual dynamical systems, so as to apply it to a wide range of system classes, from abstract models to biological and socio-economic models.

The DCI is an information theoretical measure based on the well-know Shannon Entropy (Cover and Thomas, 2006). The entropy $H(X)$ of a random variable X is computed as:

$$H(X) \; = -\sum_{x} p(x) log\, p(x) \tag{1}$$

The joint entropy of a pair of variables $H(X, Y)$ as:

$$H(X,Y) \; = -\sum_{x}\sum_{y} p(x,y) log\, p(x,y) \tag{2}$$

where Y assumes values y. Note that eq. 2 can be naturally extended to sets of k elements. As in our work we rely on observational data, probabilities are estimated by the relative frequencies of the values observed for each variable. Let us now consider a system U composed of K variables (e.g. agents, chemicals, genes, artificial entities) and suppose that S_k is a subset composed of k elements, with $k < K$. The value $DCI(S_k)$ is defined as the ratio between two measures based on the aforementioned entropy: the *integration (I)* and the *Mutual Information (MI)*. $I(S_k)$ measures the statistical independence of the k elements in S_k and is defined as:

$$I(S_k) \; = \sum_{s \in S_k} H(s) - H(S_k) \tag{3}$$

Then mutual information $MI(S_k; U \backslash S_k)$ measures the mutual dependence between subset S_k and the rest of the system $U \backslash S_k$ and it is defined as:

$$MI(S_k; U\backslash S_k) \; = H(S_k) + H(U\backslash S_k) - H(S_k, U\backslash S_k) \tag{4}$$

hence, the DCI,

$$DCI(S_k) \; = \frac{I(S_k)}{MI(S_k; U\backslash S_k)} \tag{5}$$

The value of DCI is not defined if $MI(S_k; U\backslash S_k) = 0$. However, a vanishing mutual information is a sign of separation of the investigated subset from the rest of the system, and therefore the subset must be studied separately. It is worth noting that the DCI scales with the subset size. In Villani et al. (2015) we show a procedure to normalize it, nevertheless a better approach is that of assessing the statistical significance of the DCI of S_k by means of a statistical significance index:[3]

$$t_c(S_k) \; = \frac{DCI(S_k) - \langle DCI_h \rangle}{\sigma(DCI_h)} \; = \frac{\nu DCI - \nu \langle DCI_h \rangle}{\nu \sigma(DCI_h)} \tag{6}$$

where $\langle DCI_h \rangle$ and $\sigma(DCI_h)$ are respectively the average and the standard deviation of the DCI of a sample of subsets of size k extracted from a reference system U_h randomly generated according to the frequency of each single state in U and $\nu = \langle MI_h \rangle / \langle I_h \rangle$ is the normalisation constant. It is worth noting that the aim of the reference system is that of quantify the finite size effects affecting the information theoretical measures on a random instance of a system with finite dimensions.

A list of CRS can be in principle obtained by computing the DCI of every possible subset of variables in U and ranking the subsets by DCI values. The subsets occupying the first positions are most likely to play a relevant role in system dynamics. For large-size systems, exhaustive enumeration is computationally impractical as it requires to enumerate the power set of U. In this case, we resort to heuristic algorithms, such as genetic algorithms. However, since in this work we are interested in uncovering some properties concerning the information exchanged among RS, only small systems suitable for an exhaustive assessment will be tackled, postponing to a future contribution the presentation of the application of the DCI to large-size systems.
Finally, random perturbations are applied to the dynamical system in order to observe their dynamical feedbacks also in case of stable situations.

Sieving the candidate subsets

The list of CRS can be ranked according to the significance of their DCI. In previous works we directly used this ranking to identify by hand the CRS relevant for the dynamics of the system. Nevertheless, in many cases this analysis might return a huge list of entangled CRS, so that a direct inspection is required for explaining their relevance. To this aim, we present a DCI analysis post-processing sieving algorithm to reduce the overall number of CRS to manually tackle. The main assumption of the algorithm is that if a CRS A is a proper subset of CRS B, i.e. $A \subset B$, then only the subset with the higher DCI is maintained between the two. Thus, only disjoint or partially overlapping CRSs are retained: the used assumption implies that the remaining CRSs are not further decomposable, forming in such a way the "building blocks" of the dynamical organisation of the system. The pseudo-code of the algorithm is presented in Algorithm 1.

Assessing the temporal correlation among the subsets: the D index

Although by the application of the DCI, CRS are detected, this measure does not provide indications neither on the quantity nor on the direction of the entropy, i.e. the information flowing among subsets. To this aim we applied the directionality index proposed by Lautier and Raynaud (2014).

Let X and Y be two random variables—or, equivalently, two sets of variables. We can define the **entropy rate** of X as *the average number of bits needed to encode a successive*

[3] Introduced in (Tononi et al., 1998).

Algorithm 1 Sieving algorithm

```
Input: The array $C$ of all the $CRS$ ranked by their $t_c$(DCI)
Output: A subset $RS \in CRS$
$RS \leftarrow \emptyset$
$n \leftarrow |CRS|$
Initialize auxiliary array $Del[k] \leftarrow \{0$ for $k$ in $1 \dots n\}$
for $i = 1$ to $n - 1$ do
    for $j = i + 1$ to $n$ do
        if $Del[i] \neq 1 \quad \wedge \quad Del[j] \neq 1$ then
            if $C[i] \subset C[j] \quad \vee \quad C[j] \subset C[i]$ then
                $Del[j] \leftarrow 1$
            end if
        end if
    end for
end for
for $i = 1$ to $n$ do
    if $D[i] = 0$ then
        $RS \leftarrow RS \cup \{C[i]\}$
    end if
end for
```

state of X if all the previous states are known, considering that the value of X at $t + 1$ depends either on X and Y at time t, eq. 7a, or just on the value of X at time t, eq. 7b.[4]

$$h_1 = -\sum_{X,Y} p(x^{t+1}, x^t, y^t) log\, p(x^{t+1}|x^t, y^t) \tag{7a}$$

$$h_2 = -\sum_{X,Y} p(x^{t+1}, x^t, y^t) log\, p(x^{t+1}|x^t) \tag{7b}$$

then, the transfer entropy T is defined as the difference between the aforementioned entropy rates. T describes how the uncertainty of X is reduced by knowing the previous states of Y and X itself, eq. 8.

$$\begin{aligned} T_{Y \to X} &= h_2 - h_1 \\ &= \sum_{X,Y} p(x^{t+1}, x^t, y^t)\ log\ \frac{p(x^{t+1}|x^t, y^t)}{p(x^{t+1}|x^t)}, \end{aligned} \tag{8}$$

Thus, $T_{Y \to X}$ can be described in term of entropy as:

$$T_{Y \to X} = H(X^{t+1}|X^t) - H(X^{t+1}|Y^t, X^t), \tag{9}$$

and, since the $T_{Y \to X}$ describes the information moving from Y to X, and the **transfer entropy is not symmetric**, the information from X to Y is computed as well, eq. 10.

$$T_{X \to Y} = H(Y^{t+1}|Y^t) - H(Y^{t+1}|X^t, Y^t), \tag{10}$$

Once that $T_{Y \to X}$ and $T_{X \to Y}$ are known, the directionality D of the information flow between X and Y can be measured as:

$$D_{X \to Y} = \begin{cases} 0, & \text{if } T_{X \to Y} = T_{Y \to X} \\ \frac{T_{X \to Y} - T_{Y \to X}}{T_{X \to Y} + T_{Y \to X}} \in [-1, 1], & \text{otherwise} \end{cases} \tag{11}$$

where $D_{X \to Y} = 1$ indicates that all the information moves from X to Y, i.e. absence of information flow from Y to X and, conversely, $D_{X \to Y} = -1$ indicates that X, with respect to Y, is just an information receiver and not an information sender.

It is worthwhile to notice that D does not provide any indication about the amount of information exchanged between the variables, but it only provides suitable indications on the direction of the information flow.

As introduced above, these considerations are also valid for groups of variables; however, the number of available observations limits the number of joint variables that may be analyzed (e.g. see Kraskov et al. (2004)): this is because spurious correlations are more frequently "detected" as this limit is approached, increasing the error in measurement. In order to avoid as much as possible this effect, each perturbation of the following analyses is introduced after the system has recovered a stable dynamical condition. Finally, the small size of the relevant subsets presented in this work allows a direct use of transfer entropy; in case of bigger subsets it is nevertheless possible to use sampling techniques, as in Lizier et al. (2011).

Analysis on test cases

In order to test the presented methodology, in the following we analyse several cases whose dynamical mechanisms are precisely known. We consider here boolean networks, a modelling framework that despite its apparent simplicity has obtained remarkable results in simulating real gene regulatory networks (Serra et al., 2004; Shmulevich et al., 2005; Serra et al., 2007; Villani et al., 2011). In particular, we present a system composed of 12 boolean nodes updated on the basis to either a boolean function or a random boolean value generator. Nodes update their state in parallel and synchronously. We illustrate the results of five instances of this network, defined in Table 1.[5]

[4] Note that the temporal dependency is not necessarily of unitary lag, i.e. $t - 1 \to t$. For a complete assessment of the statistical dependency of X on Y one should sum over $t-1, t-2, \dots, t-T$, where T is the observation time limit. Nevertheless, note that (i) in this paper we are analysing Markovian systems, whose behaviour depends only from the immediately previous step and (ii) although TE is not a direct measure of causal effect, the use of short history length alters the character of the measure towards inferring causal effect (Lizier and Prokopenko, 2010).

[5] Note that the size of these systems allows for an exhaustive enumeration of all the possible groups, allowing their complete assessment. It is worth remarking that each perturbation is introduced after the system has recovered a stable dynamical situation.

The five instances share a common structure but differ in specific dynamical organizations of some nodes. In *case 1*, we consider two integrated groups of three nodes (namely, *group A* and *group B*), by assigning at each node the XOR function of the other two nodes in the group. In this case the $sieving$ algorithm filtered the 94% of the evaluated CRS, making it possible to easily identify the subsets responsible for the dynamical organization of the system. Only the two correct CRSs have high t_c values, whereas all the other ones have t_c values lower of more than 2 orders of magnitude. Moreover, no information exchange takes place between *group A* and *group B*, as they are structurally independent.

Case 2 derives from case 1 by introducing in the first node of *group B* a further dependence from the last node of *group A*, hence introducing information transfer from *group A* to *group B*. The combination of the DCI analysis and the sieving algorithm correctly recalls the dynamical organization of the system—i.e. *group A* influences the behaviour of (a part of) *group B*. We observe that the whole *group B* (Figure 1b) was anyway ranked high w.r.t. its DCI significance, but it was discarded by the sieving algorithm because the dynamics of its first node is influenced also by *group A* and so the assessment of the whole *group B* is weakened. In general, the amount of this difference depends on the strength of the forces that influence the interface nodes and the elements interfacing different CRS can be detected by a simple comparison between the DCI analysis and the sieving algorithm outputs.

Case 3 derives from *case 2* by introducing a further dependence of the first node of *group A* from the last node of *group B*: again, the combination of the DCI and the sieving algorithm detects the interface nodes and the underlining dynamical system organization. Note that the asymmetry in transfer entropies (and therefore in D index) are due to differences in the initial conditions of the boolean network trajectories: a shift in initial conditions can change the direction of this asymmetry.[6]

In *case 4* five heterogeneously linked nodes are influenced by a triplet identical to that of *groupA*. The combination of the dynamical rules of the nodes and their initial condition makes the dynamical behaviour of the sixth node always in phase with the triplet, so our analysis individuates this quartet as the most relevant CRS. The other dynamical relations are not sufficiently strong to coordinate all the 8 nodes, nevertheless the method detects their influence by returning some masks[7] having high t_c values, partially overlapped with the leading quartet. The overlap of these masks indirectly indicates the presence of a greater group with respect to the winning quartet, but having a less evident dynamical presence (see Figure 1b).

Case 5 derives from *case 1* by adding two nodes whose dynamical behaviour directly depends on nodes of both *group A* and *group B*: these 8 nodes form therefore a group clearly different from the remaining 4 random nodes, as they are interdependent and ruled by deterministic functions. This group is identified by the plain DCI method, but the combination of DCI and sieving algorithm strikingly enlightens the interpretation of two triplets directly influencing a couple of nodes (Figure 1a).

Gene regulatory networks

In this section we show the application of our method to two models of regulatory networks: a model of mammalian cell cycle network (MCC in the following), as "booleanized" in Fauré et al. (2006)—see Table 2 for the chosen boolean model—and a model of one of the major cellular signal transduction pathways, known as the Mitogen Activated Protein Kinase (MAPK) cascade (Widmann et al., 1999). In Fauré et al. (2006) the authors provides a boolean dynamical model of the mammalian cell cycle, able to reproduce the main characteristics of the succession of molecular events leading to the reproduction of the genome of a cell and its division into two daughter cells.

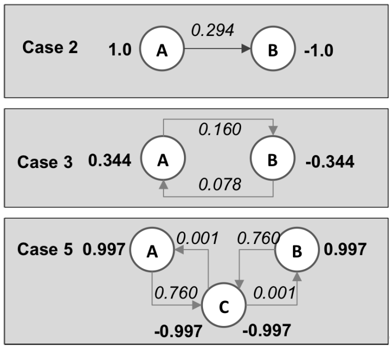
Mammalian cell division must be coordinated with the overall growth of the organism; this coordination is achieved through extra-cellular signals whose balance decides whether a cell will divide or remain in a resting state. The positive signals or growth factors ultimately elicit the activation of Cyclin D (CycD) in the cell. In the proposed model CycD thus represents the input and its activity is considered constant. By pointing the interested reader to (Fauré et al., 2006) for the details, for now it is enough to say that in absence of CycD the system presents a unique stable attractor where only Rb, p27 and Cdh1 are active, whereas in its presence E2F, CycE, CycA, Cdc20, Cdh1, UbcH10 and CycB cycle with a period of length 7. We perturbed both asymptotic states, obtaining in each case only one group (composed of E2F, CycE, CycA, Cdc20, Cdh1, UbcH10 and CycB in the first case, and of Rb, E2F, p27, Cdc20, UbcH10 and CycB in the second case). The leading groups of the two situations are different, but in each case the other CRS identified overlap with these groups and their sum cover the whole system, indicating the presence of a single coordinated pattern. So, the analysis indicates that the elements of the mammalian cell cycle network act as a single compact engine, see Figure2a.

[6] The interesting study of this effect is out of the scope of this paper and is subject of future work; in any case, a brief exploration of a limited set of initial conditions supports this preliminary claim (data not shown).

[7] CRSs can be represented by rows, where entries corresponding to variables belonging to CRS are marked in black ("masks" in the following)

Node	Node Rule				
	Case 1	**Case 2**	**Case 3**	**Case 4**	**Case 5**
N01	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)
N02	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)
N03	(N04 $\oplus$ N05)	(N04 $\oplus$ N05)	N10$\wedge$(N04 $\oplus$ N05)	(N04 $\oplus$ N05)	(N04 $\oplus$ N05)
N04	(N03 $\oplus$ N05)	(N03 $\oplus$ N05)	(N03 $\oplus$ N05)	(N03 $\oplus$ N05)	(N03 $\oplus$ N05)
N05	(N03 $\oplus$ N04)	(N03 $\oplus$ N04)	(N03 $\oplus$ N04)	(N03 $\oplus$ N04)	(N03 $\oplus$ N04)
N06	Random(0.5)	Random(0.5)	Random(0.5)	(N05 $\oplus$ N08)	(N05 $\oplus$ N08)
N07	Random(0.5)	Random(0.5)	Random(0.5)	(N07+N08+N09+N10) $\geq$ 2	$\neg$(N05 $\oplus$ N08)
N08	(N09 $\oplus$ N10)	N05$\wedge$(N09 $\oplus$ N10)	N05$\wedge$(N09 $\oplus$ N10)	N03 $\oplus$ N05	N09 $\oplus$ N10
N09	(N08 $\oplus$ N10)	(N08 $\oplus$ N10)	(N08 $\oplus$ N10)	(N04+N05+N07+N08)$\leq$ 2	(N08 $\oplus$ N10)
N10	(N08 $\oplus$ N09)	(N08 $\oplus$ N09)	(N08 $\oplus$ N09)	N06$\wedge$(N05 $\oplus$ N09)	(N08 $\oplus$ N09)
N11	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)
N12	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)	Random(0.5)

Table 1: The update rules of the boolean networks discussed on the text. Random(0.5) denotes a Bernoulli distribution with probability 0.5.



(a)

CASE 2												
N01	*N02*	*N03*	*N04*	*N05*	*N06*	*N07*	*N08*	*N09*	*N10*	*N11*	*N12*	*tC*
								■	■			**544.95**
		■	■	■								**434.08**
							■	■	■			**418.79**

CASE 3												
N01	*N02*	*N03*	*N04*	*N05*	*N06*	*N07*	*N08*	*N09*	*N10*	*N11*	*N12*	*tC*
								■	■			**858.01**
			■	■								**414.71**
							■	■	■			**393.68**
		■	■	■								**361.58**

CASE 4												
N01	*N02*	*N03*	*N04*	*N05*	*N06*	*N07*	*N08*	*N09*	*N10*	*N11*	*N12*	*tC*
		■	■	■	■							**692.53**
			■	■	■		■					**549.65**
			■	■	■				■			**524.71**
		■	■		■		■		■			**459.35**
			■	■	■	■						**420.49**

(b)

Figure 1: (a) The dynamical organization of the systems described in the text; on the edges, the Transfer Entropy, and on the nodes the D index. *Group C* in *case 5* has two values for the D index, each value depicting the *group C* role in the relation with *group A* and *group B*, respectively. (b) The CRS identified by the DCI only in some systems described in the text. Each row denotes one CRS, composed of nodes whose entries are marked in black; on the right, the value of t_c. Note that in *case 2* the third row is the correct identification of *group B*, and that in *case 3* the third and fourth rows are *group A* and *group B*. For *case 4* the output of the combination of the DCI analysis and sieving algorithm are presented; note that besides the correct group formed by nodes *N03–N06* other CRS have high t_c values, highlighting the presence of a larger but less evident dynamical group.

Product	Logical Rule leading to an activity of product	Legend
$CycD$	$CycD$	Cyclin D
Rb	$(\overline{CycD} \wedge \overline{CycE} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{CycD} \wedge \overline{CycB})$	Retinoblastoma Protein
$E2F$	$(\overline{rB} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{rB} \wedge \overline{CycB})$	Transcription factors
$CycE$	$(E2F \wedge \overline{rB})$	Cyclin E
$CycA$	$(E2F \wedge \overline{rB} \wedge \overline{Cdc20} \wedge \overline{Cdh1} \wedge \overline{Ubc}) \vee (CycA \wedge \overline{RB} \wedge \overline{Cdc20} \wedge \overline{Cdh1} \wedge \overline{Ubc})$	Cyclin A
$p27$	$(\overline{CycD} \wedge \overline{CycE} \wedge \overline{CycA} \wedge \overline{CycB}) \vee (p27 \wedge \overline{CycE} \wedge \overline{CycA} \wedge \overline{CycB} \wedge \overline{CycD})$	$p27$ enzyme inhibitor
$Cdc20$	$CycB$	Activators of the Anaphase
$Cdh1$	$(\overline{CycA} \wedge \overline{CycB}) \vee (Cdc20) \vee (p27 \wedge \overline{CycB})$	Promoting Complex
Ubc	$(\overline{Cdh1}) \vee (Cdh1 \wedge Ubc \wedge (Cdc20 \vee CycA \vee CycB))$	E2 ubiquitin conjungating enzyme
$CycB$	$(\overline{Cdc20} \wedge \overline{Cdh1})$	Cyclin B

Table 2: Adapted from Fauré et al. (2006), the boolean regulatory network of mammalian cell cycle network and a short description of each node of the system.

Matabolic pathway MAPK

The MAPK pathway (evolutionarily conserved from yeasts to humans) responds to a wide range of external stimuli, triggering growth, cell division and proliferation (Sarma and Ghosh, 2012). Sarma and Ghosh also introduce the models considered in our analysis. The basic model is composed of reactions that are quite well-established from an experimental viewpoint, and it has the hierarchical structure shown in Figure 3a. The three chemicals identified as the core of this three-layered system are the $MAPKKK$, $MAPKK$ and $MAPK$ kinases (respectively $M3K$, $M2K$ and MK for short) (Widmann et al., 1999). $M3K$ is activated by means of a single phosphorylation whereas $M2K$ and MK are both activated by double phosphorylation. Parallel to the phosphorylation by kinases, phosphatases present in the cellular volume can dephosphorylate the phosphorylated kinases (Figure 3a shows the schema of the $MAPK$ cascade where each layer of the cascade is dephosphorylated by a specific phosphatase). Note that superimposed on the three-layered structure of substrates-product reactions there is the properly called $MAPK$ signalling cascade, a linear chain of catalysis (dashed lines in Figure 3a) that transmit the external signal from $M3K^*$ to MK^{**}.[8]

When the external signal and the concentrations of the phosphatases are kept constant, a top-down reaction scheme as the one described in Figure 3a leads to fixed-point solutions. On the other hand, oscillations have been reported in the MAPK cascade (Shin et al., 2009) and, in order to account for them, Sarma and Ghosh adopt a models with feedback, one of which is described in Figure 3b. This variant (called S2 in the following) is characterized by a configuration of the activating and inhibiting interactions among layers that alters the "layered" structure of the basic model, which is no longer strictly hierarchical. This alternative model is grounded on experimental data; we will not enter here a discussion about the merits and limits of the model, referring the interested reader to the original paper, but we will take it "for granted" and we will apply our method to test whether it can discover significant dynamical organization, without any prior knowledge of the interactions, but on the sole basis of the dynamics of concentrations. We simulated the Sarma and Ghosh models with the CellDesigner software (Funahashi et al., 2008, 2003), keeping the P1, P2 and P3 phosphatases as constant (as they do) obtaining the same asymptotic states shown by the authors. In order to apply our method we perturbed the asymptotic states of these models: in particular, we focused our analysis on kinases perturbations.[9] The stable situations that are reached can show both oscillating (S2 system) or constant concentrations (basic system). Concentration changes are more significant than their absolute values (it is important to monitor the variables that are changing in coordinate way); therefore the continuous concentration values are represented according to a three levels code related to the sign of the time derivatives at time t ("decreasing concentration", "no significant change", "increasing concentration").

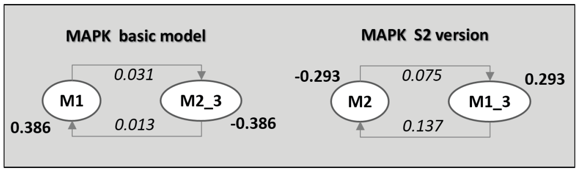


Figure 4: The dynamical organization of MAPK system (basic model and S2 version). In italic the Transfer Entropy, in bold the D index associated to the interested group within the relation. M1 group involves the first layer of 3a; M2 group involves the second layer, whereas groups M2_3 and M1_3 involve respectively the layers 2 and 3, and 1 and 3.

The combination of DCI and sieving algorithm applied to the basic MAPK model detects two dynamical groups: the first including the first layer of Figure 3a and the second including the other two layers. The two groups exchange information, the second transmitting more information to the first one (see Figure 4). The introduction of the feedbacks changes system dynamics: there are still two dynamically relevant groups, now composed of the second layer and by the other two layers, respectively. The analysis therefore suggests that the MAPK system may be decomposed in two dynamically distinct parts.

Conclusions

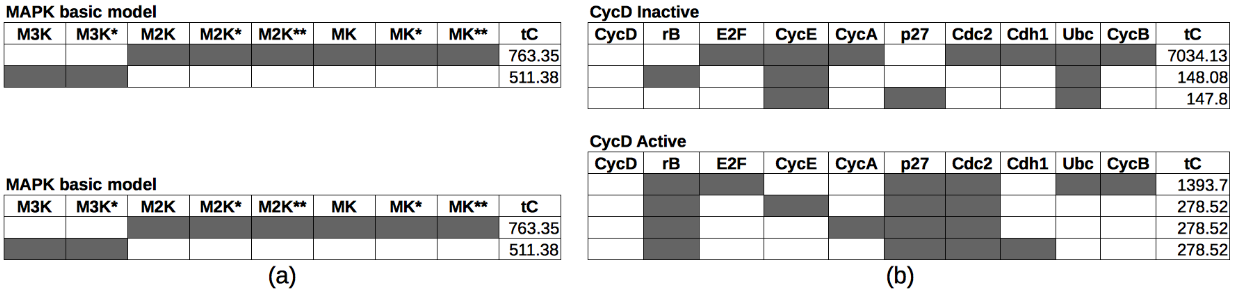
In this paper we have presented an improvement of a method based on information-theoretical measures able to identify the most relevant groups of variables that impact the dynamics of a system. We first introduced a sieving algorithm which makes it possible to identify the groups of variable that are responsible for system dynamics. Moreover, we show that it is possible to recover the direction of information flow among these groups, thus characterising the dynamical organisation of the system.

With respect to this, we noted that better normalisation methods can be applied to improve the algorithms efficacy: these methods are subject of ongoing work.

Finally, the effectiveness of this combination of techniques has been validated on test cases and subsequently applied to two prominent biological models, i.e. the mammalian cell cycle network and of Mitogen Activated Protein Kinase (MAPK) cascade.

[8] The symbol "$*$" indicates the phophorylated version of the molecule.

[9] In particular we performed 10 perturbation cycles, each cycle involving the perturbation of each single kinase and the successive relaxing to a stable situation before the subsequent perturbation, see Villani et al. (2015) for details.



MAPK basic model

M3K	M3K*	M2K	M2K*	M2K**	MK	MK*	MK**	tC
		▨	▨	▨	▨	▨	▨	763.35
▨	▨							511.38

MAPK basic model

M3K	M3K*	M2K	M2K*	M2K**	MK	MK*	MK**	tC
		▨	▨	▨	▨	▨	▨	763.35
▨	▨							511.38

(a)

CycD Inactive

CycD	rB	E2F	CycE	CycA	p27	Cdc2	Cdh1	Ubc	CycB	tC
		▨	▨	▨		▨	▨	▨	▨	7034.13
	▨		▨					▨		148.08
			▨		▨			▨		147.8

CycD Active

CycD	rB	E2F	CycE	CycA	p27	Cdc2	Cdh1	Ubc	CycB	tC
	▨	▨			▨	▨		▨	▨	1393.7
	▨		▨		▨	▨				278.52
	▨			▨	▨	▨				278.52
	▨				▨	▨	▨			278.52

(b)

Figure 2: The masks identify by the combination of the DCI and the sieving algorithm for (a) the Mitogen Activated Protein Kinase (MAPK) cascade and (b) the two dynamical conditions of the mammalian cell cycle network. In each situation only the masks having significantly high tC values are represented.

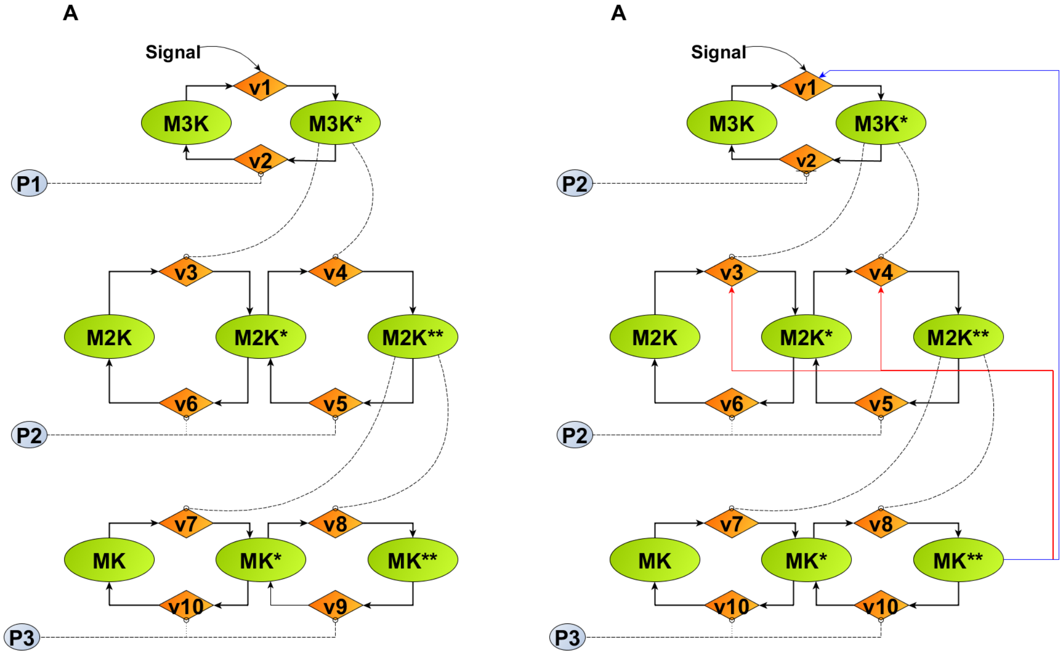


Figure 3: (A) Basic model: the scheme of the three layers MAPK cascade reaction pathway is represented (“*” stands for the phosphorylation). The signal catalyzes the phosphorylation of M3K to M3K* that in turn catalyzes the phosphorylation of M2K to M2K* and the successive phosphorylation of M2K* to M2K**. Finally M2K** performs the double phosphorylation of MK in MK** that is the final output of the MAPK cascade. P1, P2 and P3 dephosphorylate M3K, M2K and MK kinases respectively. V1-V10 stand for the involved reactions. Dashed lines with circle head represent catalysis; the figure highlights the presence of the three “layers” described on the text. (B) Two distinct positive and negative feedbacks are added to the basic model: the negative feedback goes from MK** to the second layer (M2K, M2K* and M2K**) while the positive feedback goes from MK** to the first layer (M3, M3K*).

Acknowledgements

This article has been partially funded by the UE projects "MD, Emergence by Design", Pr.ref. 284625 under the 7th FWP - FET programme. For their useful comments and suggestions, reviewers are sincerely acknowledged.

References

Chaos, A., Aldana, M., Espinosa-Soto, C., León, B. G. P., Arroyo, A. G., and Alvarez-Buylla, E. R. (2006). From Genes to Flower Patterns and Evolution: Dynamic Models of Gene Regulatory Networks. Journal of Plant Growth Regulation, 25(4):278–289.

Cover, T. and Thomas, J. (2006). Elements of Information Theory.

Farmer, J. and Kauffman, S. (1986). Autocatalytic replication of polymers. Physica D: Nonlinear Phenomena, 220:50–67.

Fauré, A., Naldi, A., Chaouiya, C., and Thieffry, D. (2006). Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle. Bioinformatics, 22(14):124–131.

Filisetti, A., Graudenzi, A., Serra, R., Villani, M., Füchslin, R. M., Packard, N., Kauffman, S. a., and Poli, I. (2011). A stochastic model of autocatalytic reaction networks. Theory in biosciences = Theorie in den Biowissenschaften, pages 1–9.

Funahashi, A., Matsuoka, Y., Jouraku, A., Morohashi, M., Kikuchi, N., and Kitano, H. (2008). CellDesigner 3.5: A versatile modeling tool for biochemical networks. Proceedings of the IEEE, 96:1254–1265.

Funahashi, A., Morohashi, M., Kitano, H., and Tanimura, N. (2003). CellDesigner: a process diagram editor for gene-regulatory and biochemical networks.

Kraskov, A., Stögbauer, H., and Grassberger, P. (2004). Estimating mutual information. Physical Review E - Statistical, Nonlinear, and Soft Matter Physics, 69(6 2).

Lautier, D. and Raynaud, F. (2014). Information flows in the term structure of commodity prices. Available at SSRN 2430168.

Lizier, J. T., Heinzle, J., Horstmann, A., Haynes, J. D., and Prokopenko, M. (2011). Multivariate information-theoretic measures reveal directed information structure and task relevant changes in fMRI connectivity. Journal of Computational Neuroscience, 30(1):85–107.

Lizier, J. T. and Prokopenko, M. (2010). Differentiating information transfer and causal effect. European Physical Journal B, 73(4):605–615.

Sarma, U. and Ghosh, I. (2012). Oscillations in MAPK cascade triggered by two distinct designs of coupled positive and negative feedback loops. BMC research notes, 5(1):287.

Schreiber, T. (2000). Measuring information transfer. Physical review letters, 85(2):461–4.

Serra, R., Villani, M., Graudenzi, A., and Kauffman, S. A. (2007). Why a simple model of genetic regulatory networks describes the distribution of avalanches in gene expression data. J Theor Biol, 246:449–460.

Serra, R., Villani, M., and Semeria, A. (2004). Genetic network models and statistical properties of gene expression data in knock-out experiments. J Theor Biol, 227:149–157.

Shin, S.-Y., Rath, O., Choo, S.-M., Fee, F., McFerran, B., Kolch, W., and Cho, K.-H. (2009). Positive- and negative-feedback regulations coordinate the dynamic behavior of the Ras-Raf-MEK-ERK signal transduction pathway. Journal of cell science, 122:425–435.

Shmulevich, I., Kauffman, S. A., and Aldana, M. (2005). Eukaryotic cells are dynamically ordered or critical but not chaotic. Proceedings of the National Academy of Sciences of the United States of America, 102:13439–13444.

Tononi, G., McIntosh, a. R., Russell, D. P., and Edelman, G. M. (1998). Functional clustering: identifying strongly interactive brain regions in neuroimaging data. NeuroImage, 7(2):133–49.

Villani, M., Barbieri, A., and Serra, R. (2011). A dynamical model of genetic networks for cell differentiation. PLoS ONE, 6.

Villani, M., Filisetti, A., Benedettini, S., Roli, A., Lane, D., and Serra, R. (2013). The detection of intermediate-level emergent structures and patterns. In Advances in Artificial Life, ECAL 2013, volume 12, pages 372–378. MIT Press.

Villani, M., Roli, A., Filisetti, A., Fiorucci, M., Poli, I., and Serra, R. (2015). The search for candidate relevant subsets of variables in complex systems. In Press on Artificial Life.

Villani, M. and Serra, R. (2013). On the dynamical properties of a model of cell differentiation. EURASIP journal on bioinformatics & systems biology, 2013:4.

Widmann, C., Gibson, S., Jarpe, M. B., and Johnson, G. L. (1999). Mitogen-activated protein kinase: conservation of a three-kinase module from yeast to human. Physiological reviews, 79(1):143–80.

Environmental bias forces parasitism in Tierra

Simon Hickinbotham   Susan Stepney

Department of Computer Science, University of York, York YO10 5GH, UK,
sjh518@york.ac.uk

Abstract

Tierra is an iconic ALife system. It has three innovative features: self-optimisation, parasitism, and an uncrashable execution environment. We have identified four sources of bias in the evolutionary dynamics of Tierra. We have run two sets of simulations: one with the original configuration of Tierra 6.02, and one with three of the biases removed. We find that the innovations observed in the original Tierra are preserved, and the debiased configuration is more amenable for future experiments with open-ended evolution.

Introduction

A key part of any evolutionary algorithm is the fitness function. This is used to drive the evolution of the system to deliver a solution to a particular problem. However, if we want a mechanism to evolve the evolutionary process (Evo-Evo), we need to examine the internal function of the system through the lens of evolution. Our approach to this is to find ways to look at the *intrinsic* fitness of a system: the relative fitness of reproducing systems that exists apart from a specific fitness function.

Many ALife systems have hidden biases encoded within them. One such system is Tierra (Ray, 1991), an ALife system with a long pedigree, feted as an open-ended evolutionary system in 1991, and classified as an *automata chemistry* in Dittrich et al. (2001). Key innovations in Tierra are the optimisation of replication (self-optimisation) and the emergence of parasitism, within an uncrashable execution environment. Tierra was one of the first computational systems to exhibit increasing complexity, placing it beyond the critical level at which the change in complexity of a self-replicator is degenerative (von Neumann, 1966, p 79-80). Despite ever-more elaborate configurations of the core Tierra framework, however, no breakthrough open-ended Tierra systems have been reported. A networked version of Tierra (Ray, 1996) was intended to enlarge the search by providing a linked Tierran meta-population, but (although it is wrong to assume a negative result if this has not been reported) the system proved difficult to manage, and faltered before any new result was identified. Related work (Ray and Hart, 2000) demonstrated differentiation in a multithreaded version of Tierra, but this involved seeding the system with a hand-coded genome ten times longer than that used in earlier studies. Thus, unbounded evolution has not been reported in Tierra. The system seems to converge to an efficient replicator that tolerates various forms of parasitism.

We hypothesise that implementational biases in ALife environments can drive the populations that inhabit them toward particular biased solutions to the problem of survival, and that this limits their capability to demonstrate continual emergent behaviours. Since several researchers have found biases in the implementation of Tierra (see below), we have an opportunity to test the hypothesis in this system.

An artificial environment is constructed from parameters, variables and mechanisms. These set up a framework in which the ALife organisms must operate. The environment does not evolve, and it sets up a drive to phenotypes that exploit it best. Biases in the environment confer non-evolvable selective advantage on individuals and can reduce the effects of competition between individuals.

Here we explore the idea that the environmental biases in Tierra may play a part in limiting the open-endedness that is observed. This counters the view that the cascade of mutations that often arises in Tierra (parasitism, 'hyper-parasitism', optimisation) represent the limits of what can evolve in automata chemistries (Channon and Damper, 1998). We have identified four biases reported in the literature. We name these as:

1. **Reaper bias**: The reaper queue mechanism preserves genomes that make no errors (Ray, 1992).

2. **Substitution bias**: Bit-flip mutations are arranged such that opcodes for similar functions are substituted for one another (Ray, 1996; Droop and Hickinbotham, 2011).

3. **Length bias**: The number of mutations per opcode changes as a function of the length of the individual (Droop and Hickinbotham, 2012).

4. **Zero-address bias**: The encoding of two instructions apportions special advantage to the individual at soup ad-

dress 0 (Baugh and McMullin, 2012).

To reveal the effect of the biases on evolutionary dynamics, we examine the Quantitative Non-Neutral (QNN) evolutionary activity (Droop and Hickinbotham, 2012) between the original version 6.02 of Tierra and a version of the same system with all these the biases removed. We find that both parasitism and self-optimisation decrease in configurations of Tierra that have biases removed.

Tierra

For the original overview of Tierra, see Ray (1991). A Tierra simulation consists of a set of individual "programs" existing in a "soup" of memory. An individual "creature" in Tierra is a string of opcodes, a pointer, a set of registers and a stack. The opcodes specify a sequence of computational operations that shift values between the stack and the registers in a manner similar to conventional computers. The sequence of opcodes is called the 'genome' of the individual, and each unique genome is assigned a species identifier code (`aaa0080` for example). The individuals compete for processor time, which is allocated via a control structure called a slicer. Another controller, the reaper, determines whether an individual should be deleted or not at each time step. The population of individuals exist together in an environment of memory with a particular fixed size (the 'soup'). The system is seeded with a single individual whose genome encodes a program that, when run, creates a copy of the individual. New species that arise by mutation explore the functional space adjacent to their parents, and those that are able to persist do so by exploiting sequences of instructions belonging to neighbouring genomes. The shorter an individual's instruction set is, the more quickly it can be copied. Individuals cannot read-protect their genomes, and so cannot prevent neighbouring creatures from executing their code.

Channon and Damper (1998) review a measure of evolutionary activity with respect to Tierra. They observe that the two main results of Tierra, namely parasitism and hyper-parasitism, are the result of two bit-flip mutations. This implies that the seed Tierran configuration has a hair-trigger cascade of mutations built into it, and that the emergent properties are close to being 'designed in'. Channon and Damper (1998) are particularly scathing here: "it also shows the ecological results to be somewhat more trivial than we might have first hoped".

Standish (2004) explores Tierra in the context of the *neutral evolution theory* (Kimura, 1983), and concludes that the co-evolutionary pressures in Tierra mean that neutrality is suppressed.

Ray and Xu (2001) also measure evolutionary activity in Tierra. They use a variety of measures, but are not wholly satisfied with the ability of any of them to capture evolutionary activity. Most of their statistics are based on the concept of a 'shadow' model (Bedau and Brown, 1999), which is difficult to implement in Tierra. To address this, they have devised a version of Tierra that carries out random substitutions of entire genomes when organisms are created or destroyed, and claim that this results in a system with no evolution. However, the environmental biases in Tierra have *not* been removed, and thus still influence which creatures in the shadow system are selected for removal.

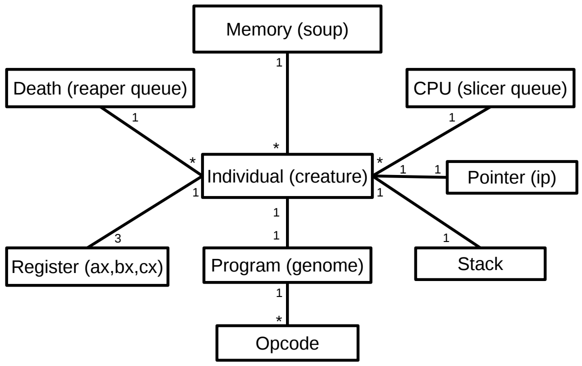


Figure 1: Platform Model of Tierra

Biases

In order to clarify our view of biases in Tierra, we have abstracted a "platform model" (Andrews et al., 2010) from the implementation, shown in figure 1. This model provides our basis for arguing about the specific biases in the implementation. It captures the relationships between the components of the system described above.

There is no specification of whether the program of an individual is 'legal' or not, but unless a program is capable of self-reproduction, then it will eventually disappear from the system due to the action of the reaper. Biases occur when the implementation of the model introduces selection pressures that are not part of the model. We detail these below.

Reaper bias: When a new individual is created, a new entry is created at the back of the reaper queue. Individuals whose entry is near the front of the queue are deleted when there is insufficient space in the memory for a new individual. Thus the chance of death increases with age. There are two occasions when an individual's order in the queue is changed. First, when certain instructions are executed in a way not intended by the designers, an error condition is raised, which bumps the individual nearer to the front of the queue, penalising it. Second, two instructions are deemed so difficult to execute that the individual is moved back down the queue if these are executed *without* error, rewarding it. So the fitness function rewards individuals whose program is 'correct'. Indeed, Harada et al. (2012) have used the Tierra reaper queue as the basis of a genetic program. This emphasises that the reaper queue has bias hard-wired into it. Yet there is no reason why a self-replicating system should not survive in the absence of these pressures, since the speed and accuracy of reproduction should be a sufficient selection pressure in itself.

Substitution bias: There are 32 instructions in Tierra, and each is represented on the genome by 5 bits. When an instruction is copied into a new genome, there is a small chance of mutation, which flips one bit, resulting in a different instruction. The bit-string encoding of instructions is arranged such that codes for similar functions are substituted for one another (Droop and Hickinbotham, 2011; Ray, 1996). Although bit-flips are common ways of carrying out mutation in genetic algorithms, in Tierra this approach introduces bias: there are only 5 opcodes that a single mutation can change a given instruction to in a single mutation (Ray, 1996). This has the effect of restricting the 'adjacent possible' of mutations. This facility is 'designed in' to Tierra (Ray, 1992, p.7), with the intention of emulating the three-base coding of amino acids in DNA. However, the replicators in Tierra are emulations of RNA world (Ray, 1992, p.5), in which translation of codons to peptides is absent. See also Baugh and McMullin (2013) for an alternative approach to implementing translation in Tierra. Since the manner of mutation is such a critical design issue, we argue that, in the absence of evidence to the contrary, it is essential that a single mutation can change one instruction on the genome to any other. This can be achieved in Tierra by setting parameter `MutBitProp = .0`. (In debiased Tierra, any other value of this parameter will have no effect.)

Length bias: The number of mutations per opcode changes as a function of the average length of the individuals in the soup (Droop and Hickinbotham, 2012). Thus, the mutation rate of the individual is dependent not just on its own properties, but also on a global property of the population. When the average length of individuals in the population is lower, there is less mutation in the system, and so it is more likely to persist. It is difficult to predict analytically the dynamics that this bias engenders, but, since parasites tend to be shorter and more numerous than genuine self-replicating individuals, this bias may favour the emergence of populations of parasites. Parasite genomes can be as little as 12 opcodes long whereas the smallest self-replicator has a length of 23 opcodes.

Zero-address bias: Baugh and McMullin (2012) have implemented von Neumann's self-reproducing automata (von Neumann, 1966) within Tierra. The resulting distributed self-reproducing system exposes a bias in the implementation of two opcodes, '`jump`' and '`return`' that were not observed when using a population of self-reproducing 'RNA-world' genomes. Those authors have found that their system tends to collapse to a situation where a 'pathological constructor' exists at address 0, which spawns offspring whose function returns their instruction pointer to the program of the parent, thus effectively giving the parent multiple CPUs to execute its code. In order to preserve the self-reproducing automata architecture, they changed the function of these opcodes so that their behaviour was not biased towards shifting the instruction pointer to address 0.

Measuring Evolutionary Activity

We are setting out to test the hypothesis that removing biases in Tierra has an influence on the evolutionary activity in the system. One problem with making this link is that the phenomenon of evolvability, also known as evolutionary activity, or the rate of evolution, is vague and difficult to measure quantitatively. Here we recap the measure of evolutionary activity first presented in Droop and Hickinbotham (2012), which gives a numeric summary of evolutionary activity and thus allows different configurations to be compared.

The reasoning behind the measure is as follows. It is assumed that a species demonstrating some new beneficial adaptation will exhibit a rapidly increasing cohort size. By contrast, a neutrally-drifting species will not show any rapid increase in cohort size. Therefore by creating a measure of cohorts that are rapidly increasing, we can detect evolutionary activity.

Let c_t^i be the number of molecules of species i a timestep t. The total cohort size at timestep t is:

$$C_t = \sum_i c_t^i \tag{1}$$

The proportion of species i at timestep t is:

$$p_t^i = c_t^i / C_t \tag{2}$$

The expected proportion of species i at timestep t is the proportion observed at the previous timestep:

$$e_t^i = \begin{cases} p_{t-1}^i & \text{if } 0 < t \\ 0 & \text{if } t = 0 \end{cases} \tag{3}$$

The activity of species i at timestep t is defined to be the square of the excess of observed over expected proportion, scaled by cohort size:

$$a_t^i = \begin{cases} \left(p_t^i - e_t^i\right)^2 & \text{if } e_t^i < p_t^i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

This definition emphasises large positive increases in cohort size for each species, particularly where this occurs within a large cohort.

The total non-neutral activity A_Q of the simulation is the sum of each species activity at each timestep:

$$A_Q = \sum_i \sum_{t=0}^{T} a_t^i \tag{5}$$

The measure A_Q has two advantages. First, it is *quantitative*: it delivers a numerical measure which allows systems to be compared. Second, it measures *non-neutral* evolutionary activity without the need for an explicit model of neutral evolution. The measure is called *Quantitative, non-neutral* (QNN) evolutionary activity. Note that QNN is applicable to any systems where changes in cohort over time can be measured. An implementation of the measure in the R programming language is available from the author's website[1].

Debiasing Tierra

We used Tierra 6.02 as our reference implementation, with two changes to the source code. First, we removed a 'divide by zero' error that occurs when the program is run on a modern processor. This happens because the time units in the original version assume that one slice of time would take more than 1 second to execute. Second, we added functionality to output population dynamics data that was amenable to analysis with our QNN software library. Neither of these changes can influence the dynamics of the system since they are merely diagnostic constructs.

We created a second "debiased" implementation of Tierra, from this reference implementation, that removes three of the biases detailed above. This has been achieved as follows:

Reaper bias: We removed all calls to the '`UpRprIf()`' and '`DownReperIf()`' functions, leaving the order of individuals in the reaper queue ranked solely in order of the time they were created.

Substitution bias: We changed the bit-flip mutation function '`mut_site`' to produce a random sequence of 5 bits whenever mutation is specified. This has the effect of randomly selecting an opcode, instead of selecting an opcode from one of the five operators with a Hamming distance of one from the instruction being copied. [2]

Zero address bias: The authors of (Baugh and McMullin, 2013) supplied us with a patch to correct the zero-address biases in the '`jump`' and '`return`' instructions. If the '`jump`' instruction has no template, it will move the instruction pointer to the address held in the Bx register. If a new creature has not yet written anything to Bx, then Bx will be zero, and the Instruction pointer will move to address 0. Similarly, the return instruction will also move the pointer to zero if the stack has not been used. In the debiased version, the '`jump`' instruction acts like a '`nop`' (no operation) instruction if there is no address template. The memory in Tierra is not circular, and Tierra does not allow individuals to span the boundary between the end of memory and the beginning. This means that it is impossible for a program to execute a set of functional instructions that would require a '`return`' to address 0. Thus it is possible to specify that any call to return the instruction pointer to address 0 acts like a '`nop`', simply incrementing the instruction pointer to the next address in the program.

The original configuration of Tierra uses a range of different mutation mechanisms. An investigation of their effect is in Droop and Hickinbotham (2012); these are left at default values in this contribution. Removing the individual length bias from each of these will be the subject of future work.

Experiments

We used the original Tierra 6.02 '`soup_in`' file to configure both systems, but made some changes to the reporting and termination parameters: `BrkupSiz = 0; DiskOut = 1; GeneBnker = 1; alive = 10000`. This ensured that we could determine the ability of each configuration to self-maintain for long periods.

We ran 100 simulations of both configurations of the system. The standard Tierra configuration completed the runs in two weeks on a cluster, whereas the debiased system completed the same number in only one week. Although it is not possible to check why there was such a large discrepancy, one explanation would be that the debiased system ran template-matching instructions (`jump`, `call` and `adr`, which are among the most processor-intensive instructions in the set) more quickly, since matches were generally found that were nearer to the instruction.

One of the features of Tierra is the massive diversity that the system generates, partly due to each creature's potential to read and execute (but not overwrite) the genomes of other individuals without any spatial or binding constraints (parameter `MemModeProt = 2`). This diversity is challenging when interpreting the results of the simulations. In the following, we have looked at the entire cohort of species in the soup, and at a subset of species whose population rose above 50 individuals in at least one of the time steps: $c_t^i > 50$ for some t. We label these groups *all species* and *dominant species* respectively. Having generated the data, we used R scripts to calculate the QNN activity for each simulation, along with other plots described below.

Results

In this section, we describe numerical outcomes of our analysis. We offer an interpretation of them in the next section.

We found that in runs with the original configuration, seven of the 100 runs ceased replicating (i.e. making successful calls to the `divide` operator) and were terminated before they reached the 10^{10} instruction limit, whereas in debiased Tierra this happened only three times. This demonstrates that debiased Tierra is no less stable than Tierra 6.02 for at least $10,000$ time steps.

Figure 2 shows the distribution of QNN activity for the two configurations of Tierra for all species, and for dominant species. The raw data is shown in red, with boxplots and beanplots (Kampstra, 2008) underlaid to emphasise the distribution of QNN. Looking at all species (left), Tierra 6.02 shows a non-uniform range of behaviours with three modes of QNN activity visible in the beanplot. Debiased Tierra shows less diverse behaviour, and overall less evolutionary activity. This picture is markedly different if we plot QNN

[1] see `http://www-users.cs.york.ac.uk/~sjh/software`

[2] This could also have been achieved by setting `MutBitProp = .0`.

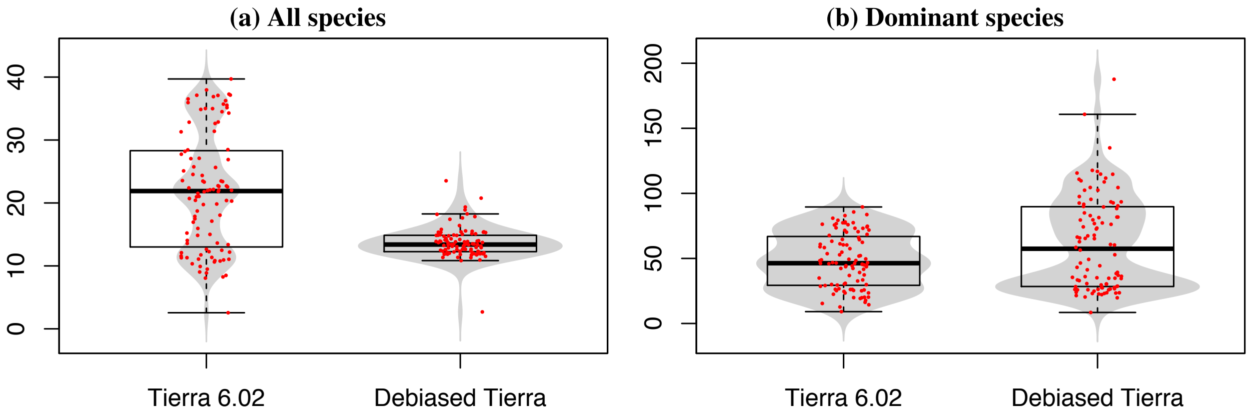


Figure 2: Beanplots plus boxplots of the distribution of QNN values (y-axis) for 100 runs of Tierra 6.02 and debiased Tierra, for (a) all species and (b) dominant species. Note the different scaling of the y-axis for figures (a) and (b).

for the dominant species (right). Here the distribution of QNN for debiased Tierra is generally higher than for Tierra 6.02. The QNN score is generally higher for the dominant species because the species proportion p for each genome type is higher if only dominant species are considered (see equation 2).

We plot the number of different species present in four simulations for each configuration of Tierra in figure 3. A feature of most simulations is that there is a period of organisation and optimisation in the system up to around $1,500$ time-steps. During this period, the population moves from a monoculture of the initial seed species to an ecosystem of different species. In both simulations, the runs with the minimum QNN score (fig. 3(a)) terminate early due to extinction of (self-) replicating entities. The median- and maximum-scoring runs are plotted in fig. 3(b) to (d). Tierra 6.02 tends to maintain a population with more species overall. The number of dominant species diminishes from around $t = 2000$. The median-scoring run for Debiased Tierra shows more volatile dynamics than the other runs. It appears to show periods of more species at around $3,000$ and $5,000$ time-steps.

An alternative way of examining the species in Tierra is to group species by genome length, as shown in figure 4. Debiased Tierra tends to produce more lengthy genomes than does Tierra 6.02. There is a greater diversity of dominant species lengths in Tierra 6.02 than in Debiased Tierra. We have examined species present at the end of each run for both configurations of Tierra. A full analysis was beyond the scope of this study, but upon brief inspection we found examples of shorter genomes that tend to behave as parasites, co-existing with longer genomes that behave as self-replicators. The organisation of genomes followed the patterns set out in Ray (1992). At the end of all runs, parasites are present in all configurations, but *dominant* species (shown in red) of parasites tend to be driven to extinction.

Our final analysis of the outputs of the simulation is to show changes in population sizes, figure 5. It is clear that debiased Tierra operates with fewer individuals overall, and fewer dominant species. Row (d) shows the maximum-scoring runs for both configurations. Tierra 6.02 shows distinct epochs in the run, where the population size changes, due to a change in the distribution of genome lengths.

Discussion

Our interpretation of the results just described is as follows.

Genome Length: The distribution of genome lengths is different between the two configurations of Tierra under study. Genome lengths tend to be longer in debiased Tierra. This is shown most clearly in figure 4: there are many more genomes that are longer (grey regions), and the dominant species also tend to have longer genomes (red regions). Since both version include length bias, we suggest that the reaper queue bias works in favour of shorter genomes: since they will replicate more quickly, they will be moved down the reaper queue more often than longer genomes.

Fixed memory size: In both configurations examined here, the memory (or 'soup') could hold $60,000$ instructions. Since longer genomes require more memory, populations of longer genomes tend to be smaller than populations of smaller genomes. This is illustrated in figure 5, where the total populations (black) are lower for debiased Tierra, particularly for runs with higher QNN (rows (c) and (d)).

The effects of a fixed memory size are echoed in the QNN scores for all species in figure 2, where Tierra 6.02 displays more evolutionary activity than debiased Tierra. The lower total population sizes for debiased Tierra limit the evolutionary activity. However, if we look only at the *dominant* species, figure 2(b), we see that several debiased Tierra runs demonstrate more innovation than any Tierra 6.02 run even though there are fewer individuals in the system.

Number of species: Debiased Tierra tends to maintain a smaller number of dominant species. By contrast, many of the runs in Tierra 6.02 display a slow reduction in the num-

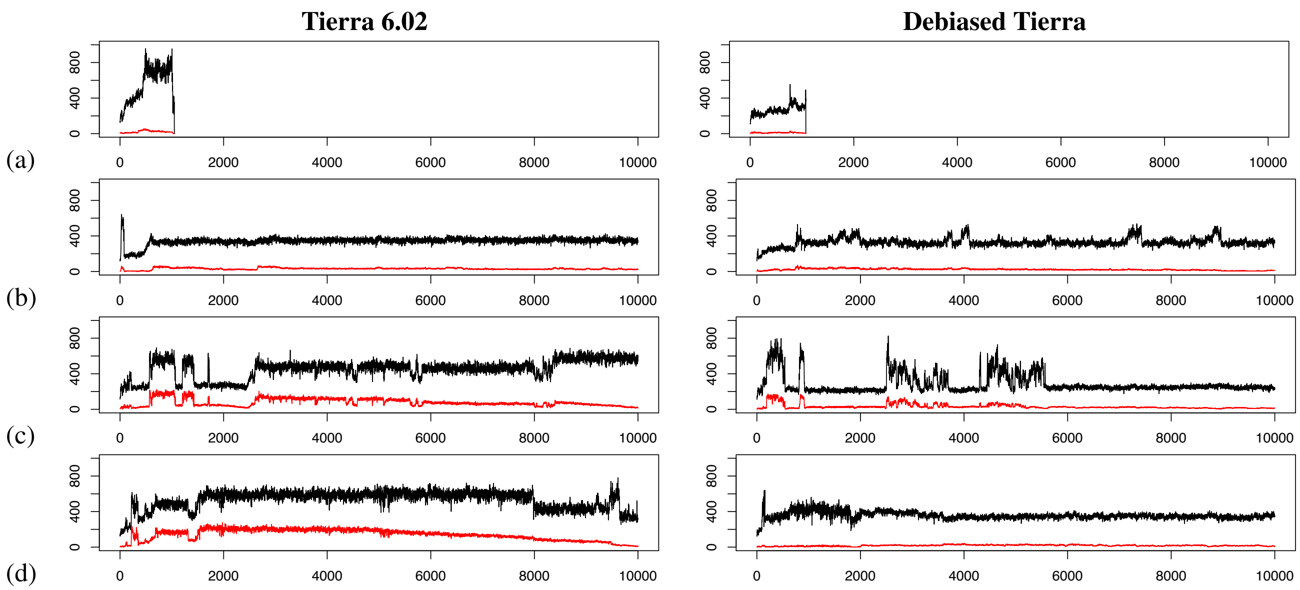


Figure 3: Four selected runs for the two configurations of Tierra, plotting the total number species (black) and the number of dominant species (red) (y-axis), against Tierran time-steps, or 10^6 instruction executions (x-axis). (a) run with the minimum QNN activity score (9.1 for Tierra 6.02 and 8.5 for Debiased Tierra);(b) run with QNN ≈ 46, the median score for the dominant species of Tierra 6.02; (c) run with QNN ≈ 53, the median score for the dominant species of debiased Tierra; (d) run with the maximum QNN activity score (89.5 for Tierra 6.02 and 187.7 for Debiased Tierra).

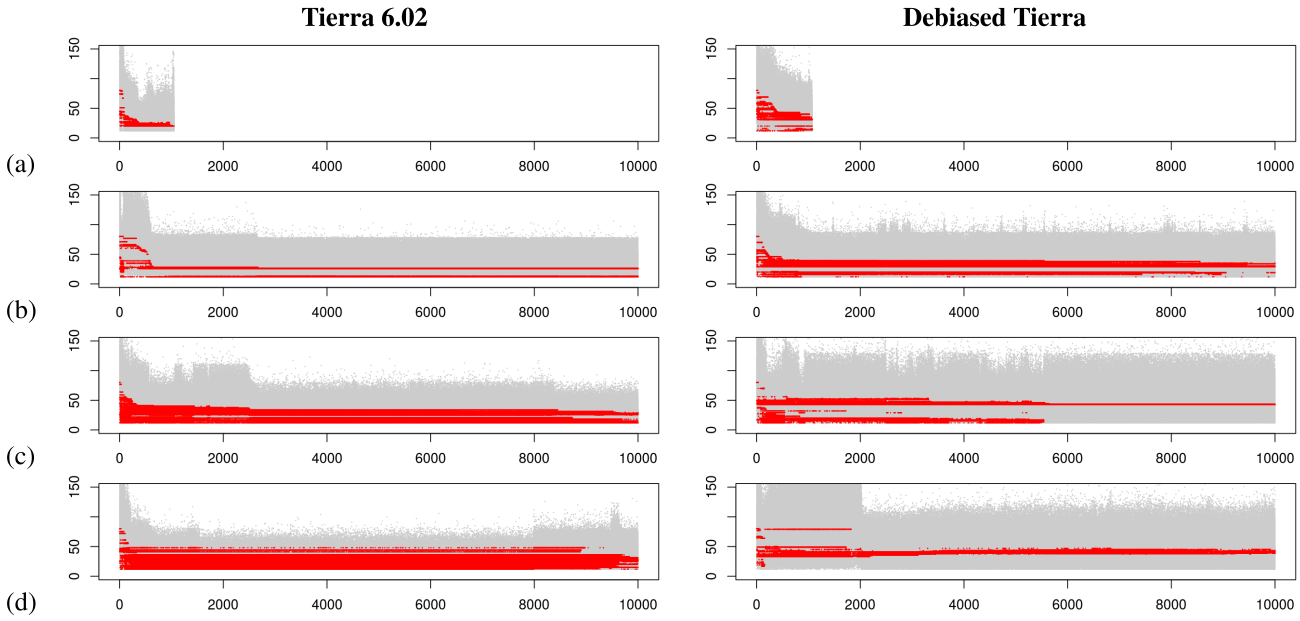


Figure 4: Four selected runs for the two configurations of Tierra, showing presence of genome lengths (y-axis) over all species (grey) and dominant species (red), against Tierran time-steps, or 10^6 instruction executions (x-axis). Rows (a) to (d) are as in figure 3.

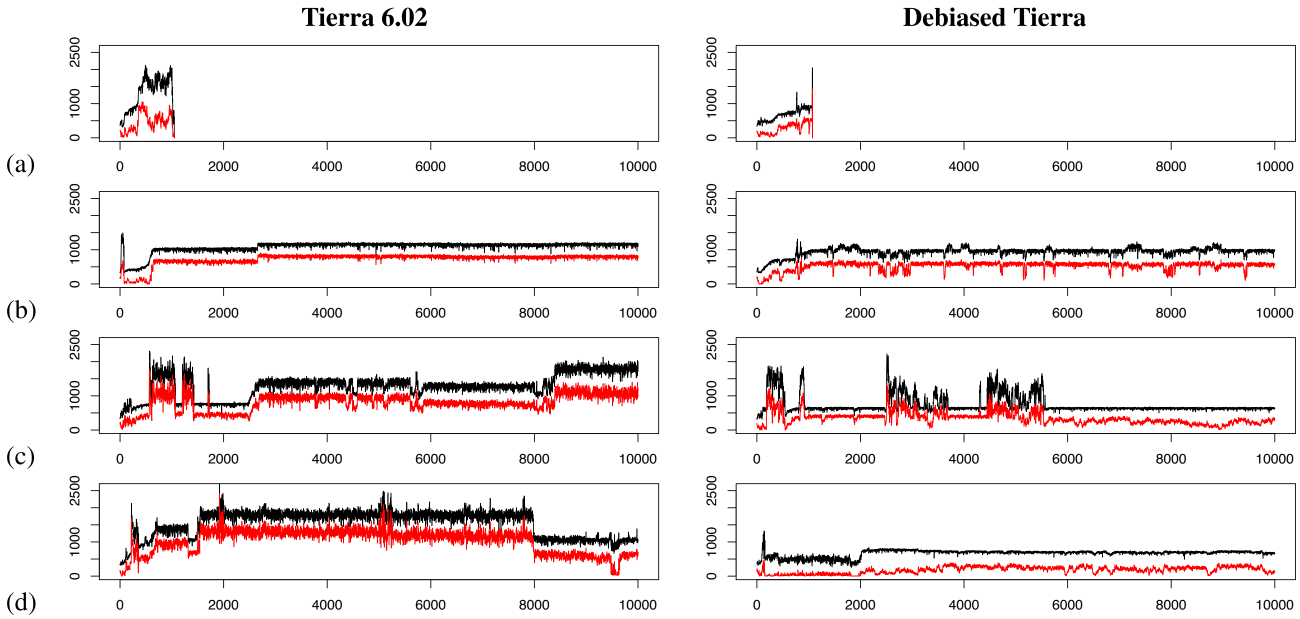


Figure 5: Four selected runs for the two configurations of Tierra, plotting population size of all species (black) and dominant species (red) (y-axis), against Tierran time-steps, or 10^6 instruction executions (x-axis).

ber of dominant species after $t = 2,000$. These results tally with the notion that genomes in Tierra 6.02 endure more parasitism, allowing mutants to emerge with low population sizes. The decline of dominant species (red lines) in figure 3 might simply be due to increasing diversity in all systems, indicating an environment where it is impossible for a particular species to dominate.

Fault Tolerance: It can be argued that manipulating the reaper queue to remove genomes that are using instructions for the purpose which they were designed endows the system with a form of fault tolerance. Interestingly, parasitic genomes operate without raising any error flags. Reaper queue manipulations act as a fitness function, biasing the system towards using instructions in the manner for which they were designed. However, after removing reaper bias, fault tolerance has increased. Evidence for this is: fewer runs terminate early; longer, less efficient genomes persist; total population is stable. Investigations into changes in diversity in debiased Tierra will be the subject of future work.

Evolutionary activity: Despite the increase in QNN activity for debiased Tierra over Tierra 6.02, these experiments could be interpreted as a negative result for debiased Tierra, since no new emergent behaviour evolves. However, we note that the debiased configuration reduces the selection pressure for short genomes, allowing *recovery from*, or *resistance to* parasitism to emerge more frequently and more strongly. In addition, there is no reduction in survival of a self-replicating cohort. We suggest that the main result here is that *the innovations observed in the original Tierra are preserved*, and *the debiased configuration is more amenable for future experiments with open-ended evolution.*

Further work

These experiments have addressed some issues with Tierra that have been reported in the last quarter of a century. In this paper we have studied whether a debiased system would yield measurable differences: it does. Having established this, next we need to determine *which* of the biases has the most effect. Our analysis revealed that the biases placed selective advantage on parasitism and shorter genomes due to the fixed memory size of the environment. We have suggested mechanisms by which length bias and reaper bias could effect the dynamics of Tierra 6.02. We can offer no interpretation of the effect that the other two changes we made, namely removing substitution bias and zero-address bias. Further work is needed to investigate their effects. Indeed, the effects of these biases may only be revealed with a different self-replicating cohort of Tierran species, as shown in the work of Baugh and McMullin (2013).

Even though Tierra has some tools for analysis of runs, the analysis of Tierran genomes is a time-consuming affair, and it was not possible to complete an analysis of the interactions between all the dominant species in all 100 runs of each configuration for this study. Tools to automate this analysis offer the potential to help the search for innovation in automata chemistries generally.

Even in a debiased configuration, it might not be possible for longer genomes to generate as much QNN as shorter ones, simply because fewer of them can fit in the memory: they cannot possibly increase in numbers as much as shorter ones can. This is a problem with having a fixed carrying capacity of the soup.

Restricting the carrying capacity ensures that there is never an unbounded population explosion in Tierra runs. There are alternative means of preventing such an explosion. For example, we have recently experimented with conservation of matter in the Stringmol automata chemistry (Hickinbotham and Stepney, 2015), and shown that it can limit population size whilst having a positive effect on evolutionary activity.

Finally, it may be possible to use Tierra in a framework where we use QNN as a fitness function to evolve a population of Tierran systems with increasing evolutionary activity. Tierra is particularly amenable to this, since it automatically records its terminal configuration in a text file called 'soup_out' that can be used as the input to a new simulation. This approach has analogies with the work of Gutierrez et al. (2014), where an activity measure was applied to a chemical system to evolve novel configurations of catalysts.

The original Tierra tackled many problems at once, so it is unsurprising that there are a few artefacts in the design that merit improvement. It is difficult to determine what the best improvements are without a numerical measure of evolutionary activity. With QNN to hand, we have been able to test our hypothesis regarding biases in the original configuration. This approach opens the door to exploring and comparing the respective merits of a range of automata chemistries, with the goal of refining and consolidating the designs in order to address the problem of generating open-ended evolution.

Acknowledgements

We thank Baugh and McMullin for supplying their zero-address bias patch, and the reviewers for their comments and suggestions. The experiments were performed using the Volvox cluster at the York Centre for Complex Systems Analysis. Part of this work was funded by the EU FP7 project EvoEvo, grant number 610427.

References

Andrews, P. S., Polack, F. A. C., Sampson, A. T., Stepney, S., and Timmis, J. (2010). The CoSMoS process, version 0.1: A process for the modelling and simulation of complex systems. Technical Report YCS-2010-453, Department of Computer Science, University of York.

Baugh, D. and McMullin, B. (2012). The emergence of pathological constructors when implementing the von neumann architecture for self-reproduction in tierra. In *From Animals to Animats 12*, volume 7426 of *LNCS*. Springer.

Baugh, D. and McMullin, B. (2013). Evolution of GP mapping in a von Neumann self-reproducer within Tierra. In *Proc. ECAL 2013*, volume 12, pages 210–217.

Bedau, M. A. and Brown, C. T. (1999). Visualizing evolutionary activity of genotypes. *Artificial Life*, 5(1):17–35.

Channon, A. D. and Damper, R. I. (1998). Perpetuating evolutionary emergence. In *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*, pages 534–539. MIT Press.

Dittrich, P., Ziegler, J., and Banzhaf, W. (2001). Artificial chemistries – a review. *Artificial Life*, 7(3):225–275.

Droop, A. and Hickinbotham, S. (2011). Application of small-world mutation topologies to an artificial life system. In *Proc. Ecal 2011*, pages 208–209. MIT Press.

Droop, A. P. and Hickinbotham, S. J. (2012). A quantitative measure of non-neutral evolutionary activity for systems that exhibit intrinsic fitness. In Adami, C., Bryson, D. M., Ofria, C., and Pennock, R. T., editors, *Artificial Life 13*. MIT Press.

Gutierrez, J. M. P., Hinkley, T., Taylor, J. W., Yanev, K., and Cronin, L. (2014). Evolution of oil droplets in a chemorobotic platform. *Nature communications*, 5.

Harada, T., Ichikawa, Y., and Takadama, K. (2012). Evolving conditional branch programs in tierra-based asynchronous genetic programming. In *Joint 6th International Conference on Soft Computing and Intelligent Systems (SCIS) and 13th International Symposium on Advanced Intelligent Systems (ISIS)*, pages 1023–1028. IEEE.

Hickinbotham, S. and Stepney, S. (2015). Conservation of matter drives open-ended evolution. In *Proc. Ecal 2015 (in press)*. MIT Press.

Kampstra, P. (2008). Beanplot: A boxplot alternative for visual comparison of distributions. *Journal of Statistical Software, Code Snippets*, 28:1–9.

Kimura, M. (1983). *The Neutral Theory of Molecular Evolution*. Cambridge University Press.

Ray, T. S. (1991). An approach to the synthesis of life. In C. Langton, C. Taylor, J. D. Farmer, S. Rasmussen, editor, *Artificial Life II, Santa Fe Institute Studies in the Sciences of Complexity*, volume vol. XI, pages 371–408. Addison-Wesley.

Ray, T. S. (1992). Evolution, ecology and optimization of digital organisms. Technical report, Santa Fe Institute Working Paper 92-08-O42.

Ray, T. S. (1996). Network Tierra report. Technical report, University of Oklahoma. http://life.ou.edu/tierra/netreport/.

Ray, T. S. and Hart, J. F. (2000). Evolution of differentiation in multithreaded digital organisms. *Artificial Life*, 7:132–140.

Ray, T. S. and Xu, C. (2001). Measures of evolvability in Tierra. *Artificial Life and Robotics*, 5(4):211–214.

Standish, R. (2004). Tierra's missing neutrality: case solved. In Pollack, J., editor, *Proceedings Artificial Life IX*, pages 364–368.

von Neumann, J. (1966). *Theory of Self-Reproducing Automata*. University of Illinois Press.

Distributed vs. Centralized Particle Swarm Optimization for Learning Flocking Behaviors

Iñaki Navarro, Ezequiel Di Mario and Alcherio Martinoli

Distributed Intelligent Systems and Algorithms Laboratory,
School of Architecture, Civil and Environmental Engineering,
École Polytechnique Fédérale de Lausanne
{ezequiel.dimario, inaki.navarro, alcherio.martinoli}@epfl.ch

Abstract

In this paper we address the automatic synthesis of controllers for the coordinated movement of multiple mobile robots. We use a noise-resistant version of Particle Swarm Optimization to learn in simulation a set of 50 weights of a plastic artificial neural network. Two learning strategies are applied: homogeneous centralized learning, in which every robot runs the same controller and the performance is evaluated externally with a global metric, and heterogeneous distributed learning, in which robots run different controllers and the performance is evaluated independently on each robot with a local metric. The two sets of metrics enforce Reynolds' flocking rules, resulting in a good correspondence between the metrics and the flocking behaviors obtained. Results demonstrate that it is possible to learn the collective task using both learning approaches. The solutions from the centralized learning have higher fitness and lower standard deviation than those learned in a distributed manner. We test the learned controllers in real robot experiments and also show in simulation the performance of the controllers with increasing number of robots.

Introduction

This article tackles the synthesis of high-dimensional controllers for cooperative tasks performed by resource-constrained robots. Evaluative machine-learning techniques are an alternative to model-based control design that may allow for full exploitation of the platforms' limited sensing capabilities, coping with discontinuities and nonlinearities, as well as dealing with noise in the performance evaluations (Floreano and Mondada, 1996; Baldassarre et al., 2007; Gauci et al., 2014; Jin and Branke, 2005; Pugh and Martinoli, 2009).

As in our previous work (Di Mario et al., 2014b), the cooperative task chosen is a loosely-coordinated collective movement or flocking (Balch and Arkin, 1998; Olfati-Saber, 2006; Antonelli et al., 2008; Navarro and Matía, 2011), in which a group of robots move together. Some researchers have previously shown that it is feasible to use learning to generate cooperative behaviors (Matarić, 2001; Parker, 1997; Baldassarre et al., 2007; Gauci et al., 2014). Matarić (2001) and Parker (1997) addressed the topic of learning in multi-robot teams using a small number of parameters per robot, as opposed to the large search space considered in this paper. It should be noted that the task as implemented in this article is harder than those presented in other contributions as the robots are not physically connected to each other (Baldassarre et al., 2007), they are required not only to aggregate but also move together (Gauci et al., 2014), and there is no environmental template or goal to guide their movement (Floreano and Mondada, 1996). Finally, in the case of Baldassarre et al. (2007) and Gauci et al. (2014) learning has been done only in a centralized manner, using homogeneous controllers and a global performance metric.

Morihiro et al. (2006) used Q-learning to generate flocking behaviors of virtual agents (not robots) in the presence of a predator, where the agents individually learn discrete actions similar to Reynolds' rules.

Some researchers have used different optimization techniques to improve the performance of manually designed flocking controllers, using PSO (Lee and Myung, 2013; Etemadi et al., 2012), gradient descent (Chang et al., 2013), Reinforcement Learning (Hayes and Dormiani-Tabatabaei, 2002), or Evolutionary Strategies (Celikkanat, 2008). Our approach in this article differs in that our behaviors are generated by a highly plastic artificial neural network and not by a specific control design targeted to flocking behavior. In other words, the main goal of this article is to compare centralized and distributed learning methods for design and optimization of collaborative behaviors, among which flocking has been chosen as a benchmark.

The distributed learning evaluates several candidate solutions in parallel on the available robotic resources. Such an approach allows the distributed robotic system to increase its robustness to failure of individual robots and speed up the overall learning process (Di Mario and Martinoli, 2014b). In order to compare the distributed and centralized approaches, we aim to design a pair of global and local fitness functions that result in the desired flocking behavior. The local or individual metric must be evaluated locally by each robot and be close to the global metric.

The second aim of this paper is to get additional correspondence between the fitness metric used and the flocking

Figure 1: Four Khepera III robots performing one of the learned flocking algorithms presented in this article.

behavior observed, in particular in respect to our previous work (Di Mario et al., 2014b). In order to achieve it, we augmented the fitness metrics to enforce the three Reynolds' flocking rules (Reynolds, 1987), by adding alignment with neighboring flockmates to the originally implemented collision avoidance and attraction. As a consequence of a better alignment and therefore tighter motion coordination, the local and global performances match better.

The remainder of this article is organized as follows. In the next section we describe the robotic platform, learning algorithms, fitness metrics and control architecture. In Section Experimental Results and Discussion, we present the different experiments performed and discuss the results obtained both in simulation and with real robots. Finally, the last section draws the conclusions of this work and discusses the limitations of the approach.

Methodology

A variation of Particle Swarm Optimization (PSO) (Kennedy and Eberhart, 1995) is used in this article in order to learn flocking behaviors. The learning problem for PSO is choosing a set of parameters of an underlying robotic controller such that a given fitness metric is maximized. The learning process is performed completely in simulation, while the learned solutions are tested both using high-fidelity simulation and real robots.

Experimental Platform

The experimental platform used is the Khepera III mobile robot, a differential wheeled vehicle with a diameter of 12 cm (see Fig. 1). Its sensing capabilities are augmented with a relative positioning system (Pugh et al., 2009), which calculates range and bearing to nearby robots based on the strength of an infrared signal. The system also communicates the ID of the robot, allowing to estimate also the heading of neighboring robots by exchanging the bearings between a pair of robots. In our experiments this communication is done using the IEEE 802.11 wireless standard and UDP messages. The Khepera III has two wheel encoders, which are used to estimate the trajectory followed by the robots for the local fitness calculations.

Simulations are performed in Webots (Michel, 2004), a high-fidelity submicroscopic simulator that models dynamical effects such as friction and inertia. In this context, by submicroscopic we mean that it provides a higher level of detail than usual microscopic models, faithfully reproducing intra-robot modules (e.g., individual sensors and actuators). The simulator has a built-in relative positioning system that gives information about the distance and direction to neighboring robots within line-of-sight, mimicking the one used in the real robots.

```
Initialize particles
for $N_i$ iterations do
    for $N_p$ particles do
        Update particle position
        Evaluate particle
        Re-evaluate personal best
        Aggregate with previous best
        Share personal best
    end for
end for
```

Figure 2: Noise-resistant PSO algorithm.

Learning Algorithm

The PSO algorithm used is a noise-resistant version introduced by Pugh et al. (2005). It works by re-evaluating personal best positions and aggregating them with the previous evaluations, in our case by performing a regular average at each iteration of the algorithm. The pseudocode for the algorithm is presented in Fig. 2.

Each particle position represents a set of parameters of the controller. As defined in Eq. 1, the movement of particle i in dimension j depends on three components: the velocity at the previous step weighted by an inertia coefficient w, a randomized attraction to its personal best $x^*_{i,j}$ weighted by w_p, and a randomized attraction to the neighborhood's best $x^*_{i',j}$ weighted by w_n. $rand()$ is a random number drawn from a uniform distribution between 0 and 1.

$$v_{i,j} = w \cdot v_{i,j} + w_p \cdot rand() \cdot (x^*_{i,j} - x_{i,j}) + w_n \cdot rand() \cdot (x^*_{i',j} - x_{i,j}) \tag{1}$$

Using the PSO algorithm we explore two different learning schemes, characterized by the way the particles are distributed among the robots and the fitness function used. The first, *global homogeneous*, copies the same candidate solution (or set of weights) to every robot, and uses a global fitness function that evaluates the group behavior. The second, *local heterogeneous*, distributes a different candidate solution to each robot, and uses a local fitness function that is evaluated independently and individually on each robot. The distributed version allows to speed up the evaluations by a factor equal to the number of robots.

The PSO neighborhood is implemented as a ring topology with one neighbor on each side. Particles' positions and velocities are initialized randomly with a uniform distribution in the $[-20, 20]$ interval, and their maximum velocity

Table 1: PSO parameter values

Parameter	Value
Number of robots N_{rob}	4
Swarm size N_p	52
Iterations N_i	200
Evaluation span t_e	4x45 s
Re-evaluations N_{re}	1
Personal weight w_p	2.0
Neighborhood weight w_p	2.0
Dimension D	50
Inertia w	0.8
V_{max}	20

is also limited to that interval. The PSO algorithmic parameters (see Table 1) are set following the guidelines for limited-time adaptation presented in our previous work (Di Mario and Martinoli, 2014a). These guidelines recommend a swarm size equal to the dimension of the search space. Since the dimension is 50 and four robots are used, we round up the swarm size to 52 particles in order to have exactly 13 particles per robot in the distributed implementation.

It is worth noticing that in the distributed heterogeneous learning, groups of four particles are always evaluated together as a flock of four robots. In these groups of four particles, two of them have a PSO neighborhood of particles from the same group, while each of the other two share their neighborhood with one particle of another group and one from its group. When testing the best controller from a distributed heterogeneous learning, we find the particle with the best local performance and test it together with the other three particles of its group.

Fitness Functions

In this section, we define the fitness functions used for centralized and distributed learning, using a global metric for the first and a local metric for the second. Both performance functions have three factors: movement, alignment, and compactness. These factors reward robots that move as far as possible from their initial positions, align their headings, and stay close to each other without colliding. The factors are all normalized to the interval $[0,1]$.

In the real experiments, all positions and distances used in the global performance metric are obtained with a global tracking system that can detect all robots at any given time, using an overhead camera connected to a computer running SwisTrack (Lochmatter et al., 2008).

The movement factor of the global performance metric (f_{1g}) is the normalized distance between the initial and the final positions of the center of mass of the group of robots. The normalization factor is the maximum distance that a robots can travel in one evaluation, i.e., the robot's maximum speed multiplied by the evaluation time.

$$f_{1g} = \frac{|\vec{x}_c(t_f) - \vec{x}_c(t_0)|}{D_{max}} \tag{2}$$

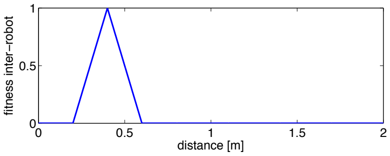


Figure 3: Inter-robot fitness as a function of the distance between two robots.

The global alignment factor (f_{2g}) quantifies the heading difference between two robots (H_{diff}) averaged between every pair of robots and during the evaluation time. It has a maximum value of 1 when all the robots are aligned and tends to 0 when robots are not aligned. It is defined as:

$$f_{2g} = 1 - \frac{1}{N_{eval}} \sum_{k=1}^{N_{eval}} \left(\frac{1}{N_{pairs}} \sum_{j=1}^{N_{pairs}} abs(H_{diff_{j,k}})/\pi \right) \tag{3}$$

where N_{eval} is the number of time steps in the evaluation period, N_{pairs} is number of inter-robot pairs and $H_{diff_{j,k}}$ is the difference of heading between pair j at time step k. Note that if there are more than two robots its value can never be 0.

The global compactness factor (f_{3g}) is the average over the evaluation time and over each pair of robots of the inter-robot fitness. We define the inter-robot fitness between two robots as a function of the distance between them, as shown in Fig. 3. The fitness is maximum at the desired inter-robot distance of $0.4m$, and it is zero when the robots are closer than $0.2m$ (slightly larger than the robots' diameter) or further apart than $0.6m$. It rewards robots that stay close to each other without colliding, implementing two of the Reynolds' rules. At each time step, we calculate the inter-robot fitness for each pair of robots, and then average across all pairs:

$$f_{3g} = \frac{1}{N_{eval}} \sum_{k=1}^{N_{eval}} \left(\frac{1}{N_{pairs}} \sum_{j=1}^{N_{pairs}} fit_{inter_{j,k}} \right) \tag{4}$$

where $fit_{inter_{j,k}}$ is the inter-robot fitness for inter-robot pair j at time step k.

The local performance metric is calculated individually by each robot, using exclusively on-board resources and mimicking the global metric. The local movement factor (f_{1l}) is defined in two different variations. The first is the normalized distance traveled by the robot (f_{1al}), based on the final position, which is calculated with odometry using the wheel encoders. The second is the normalized distance traveled by the center of mass of the group of robots (f_{1bl}), calculated using the odometry of the robot and the relative position to neighboring robots. If a neighboring robot position can not be estimated (due to occlusions or limited range of the relative positioning system), the last absolute position where the robot was seen is used as final position.

$$f_{1al} = \frac{|\vec{x}_i(t_f) - \vec{x}_i(t_0)|}{D_{max}} \tag{5}$$

$$f_{1bl} = \frac{|\vec{x}_c(t_f) - \vec{x}_c(t_0)|}{D_{max}} \tag{6}$$

f_{1bl} matches the global movement factor better than f_{1al}, but tends to evaluate all the particles in the group of robots with a very similar performance, although the controllers could be very different. Two different local metrics are employed on this article depending on the local movement factor used.

The local alignment factor (f_{2l}) is equivalent to the global alignment factor measuring the absolute heading difference between pairs of robots as in Eq. 3. The difference here is that each robot calculates its own metric only measuring the heading difference between itself and the other three robots, using the relative positioning system and communication. These measurements might be affected by occlusions and range limitations. If for a time step no neighbor is seen then H_{diff} is set to 1 for that time instant.

The local compactness factor (f_{3l}) is implemented as in Eq. 4, based on the inter-robot fitness. However, in the local metric the number of pairs N_{pairs} in Eq. 4 is modified so that each robot only measures the distance to the other three using the relative positioning system and then averages the inter-robot fitness only for those other three robots, as opposed to averaging across all pairs of robots. Another difference between the local and global compactness factors is that the local inter-robot distance measurements are affected by occlusion, while the global ones are not.

Both global and local fitness are obtained by aggregating the three corresponding factors using a generalized aggregation function described by Zhang et al. (2008):

$$F = \left(\frac{\omega_1 f_1^s + \omega_2 f_2^s + \omega_3 f_3^s}{\omega_1 + \omega_2 + \omega_3} \right)^{\frac{1}{s}} \tag{7}$$

where f_i are the individual fitness factors (with $f_i = f_{il}$ for the local fitness, and $f_i = f_{ig}$ for the global), ω_i their corresponding aggregation weights, and s is the degree of compensation. We set $s = 0$, i.e., the highest degree of compensation in design-appropriate aggregation functions, simplifying Eq. 7 to:

$$F = \lim_{s \to 0} \left(\frac{\omega_1 f_1^s + \omega_2 f_2^s + \omega_3 f_3^s}{\omega_1 + \omega_2 + \omega_3} \right)^{\frac{1}{s}} = \left(f_1^{\omega_1} f_2^{\omega_2} f_3^{\omega_3} \right)^{\frac{1}{\omega_1 + \omega_2 + \omega_3}} \tag{8}$$

Since the three factors (f_i) are in the interval $[0,1]$, the fitness function F will also be in the same range. The aggregation weights used are: $\omega_1 = 0.4$, $\omega_2 = 0.5$, and $\omega_3 = 0.1$.

In our previous work (Di Mario et al., 2014a), we showed that the fitness evaluations for learning a simpler robotic task had a large standard deviation, and that performing re-evaluations was an effective way of dealing with this challenge in the learning. Given the more complex behavior to be learned in this article and the difficulties encountered while doing so, we decided to perform multiple internal evaluations of the fitness and average them in order to make the learning more robust. Concretely, each candidate solution is evaluated four times during 45 s and its performance averaged ($F' = \frac{1}{4} \sum_{i=1}^{4} F_i$) before consideration by the noise-resistant algorithm shown in Fig. 2.

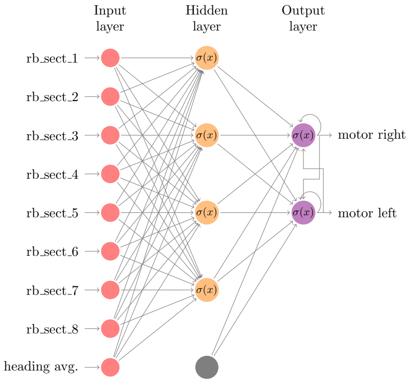


Figure 4: Diagram of the neural network controller. In red are the inputs, yellow the hidden layer with sigmoidal outputs, in blue the sigmoidal outputs which control the motor speed, and in gray the bias input.

Controller Architecture

The controller is an artificial neural network with nine inputs, a hidden layer of four units with sigmoidal activation functions, and two output units also with sigmoidal activation (see Fig. 4). The output neurons have also as input a connection from a constant bias speed, a recurrent connection from its own output, and a lateral connection from the other neuron's output. The controller uses only local, on-board measurements regardless of the performance metric. Its inputs are the range and bearing measurements and the heading average among the robots, while the outputs determine the two wheel speeds. The total number of weights to be optimized by the PSO algorithm is 50. The hidden layer, not present in our previous work (Di Mario et al., 2014b), introduces additional plasticity to the controller.

The eight range and bearing inputs (*rb_sect_k*) are obtained by dividing the bearing into eight sectors, and calculating the activation of each sector by taking the minimum range value measured in that sector and dividing it by the maximum possible range, which is 3.3 meters. The ninth input corresponds to the average of the headings among all the neighboring robots, in the robot's own coordinate system and normalized to the interval [-1,1]. The use of a single averaged input instead of one input per robot allows the controller to generalize to any number of robots.

Experimental Results and Discussion

The learning process is performed completely in simulation. We run three different optimization sets depending on the learning schema and fitness function used: *global homogeneous* (centralized) learning, *local heterogeneous* (distributed) learning with *individual* movement factor, and *local heterogeneous* (distributed) learning with *group* movement factor. Since PSO is a stochastic optimization method, we perform 20 optimization runs for each of these learning schemes.

Each evaluation during the learning process has a duration of 45 s and takes place in an unbounded arena. Four robots are placed forming a square of side length equal to two robot diameters with random orientations. The local fitness function is calculated by the robots using only their internal measurements (simulated range and bearing and wheel encoders, both with added noise), while the global fitness function is calculated using the robots' global positions with no errors provided by the simulator.

The learning progress is shown in Fig. 5 for the three learning sets, representing the best solution found at each iteration for the three different learning approaches. The curves show the average of the 20 runs, and the error bars represent the standard deviation. In the case of local heterogeneous sets it shows not only the local metric but also the global one, since it is designed to reflect the quality of the flocking behavior and allows for comparison with centralized learning.

Comparing Fig. 5a with Fig. 5b and Fig. 5c, we can see that global homogeneous learning achieves the best global metric performance and lowest standard deviation of the three methods. Also, it requires less iterations to learn as the learning curve becomes flatter faster, although the homogeneous approach employs four times the evaluation time of the heterogeneous approaches for each iteration.

In Fig. 5c (heterogeneous with group movement factor), there is a perfect matching of local and global metrics. On the other hand, in Fig. 5b (heterogeneous with individual movement factor) the local and global metrics do not match initially, but they converge to the same value as the learning progresses.

The individual movement factor is easy to learn (robots just move straight), so it achieves a high value in the initial iterations. However, because of the lack of alignment and compactness, robots spread and do not achieve the desired behavior. As the run progresses, alignment and compactness factors improve, and therefore the difference between local and global metrics is reduced. Both alignment and compactness factors still have margin for learning and might produce an improvement with further iterations.

After the learning process is finished, the fitness of the best solution from each of the 20 independent learning runs is evaluated systematically in simulation, running 100 experiments of 60 s for each solution.

From Fig. 6a we can see that homogeneous learning achieves a high performance with low standard deviation for all the runs. Both heterogeneous approaches learn the desired behavior in most runs, but sometimes fail, resulting in a high standard deviation. We noticed that this was due to two opposite reasons: in the heterogeneous learning with individual movement factor (Fig. 6b), individual speed is rewarded so the robots sometimes split. On the other hand, in heterogeneous with group movement factor (Fig. 6c) robots sometimes aggregate close to their initial positions in a very compact group and fail to travel far, resulting in low performance. This might be caused by obtaining very similar evaluations of the four particles tested together, which does not reflect the differences among the four controllers, discarding potential good solutions and promoting bad ones.

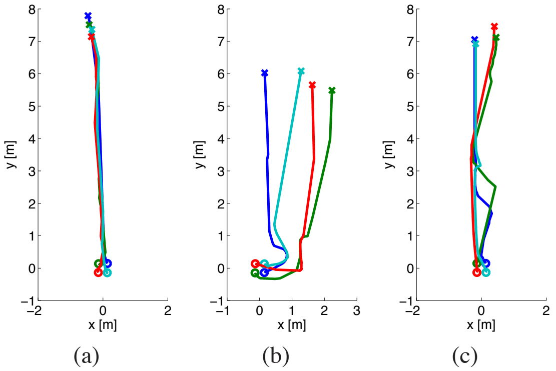
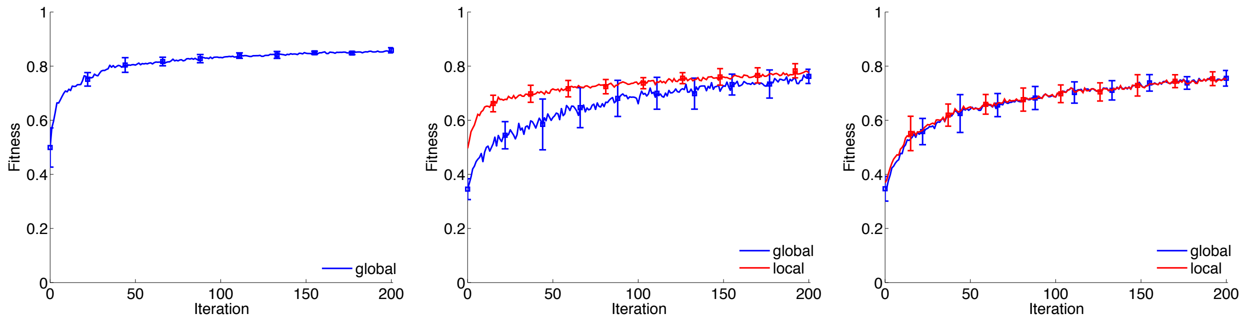


Figure 7: Example of trajectories of four robots flocking in simulation during 60 s for selected controllers from: (a) global homogeneous learning, (b) local heterogeneous with individual movement factor, and (c) local heterogeneous with group movement factor. The initial positions are marked with a circle, while the final positions are marked with a cross.

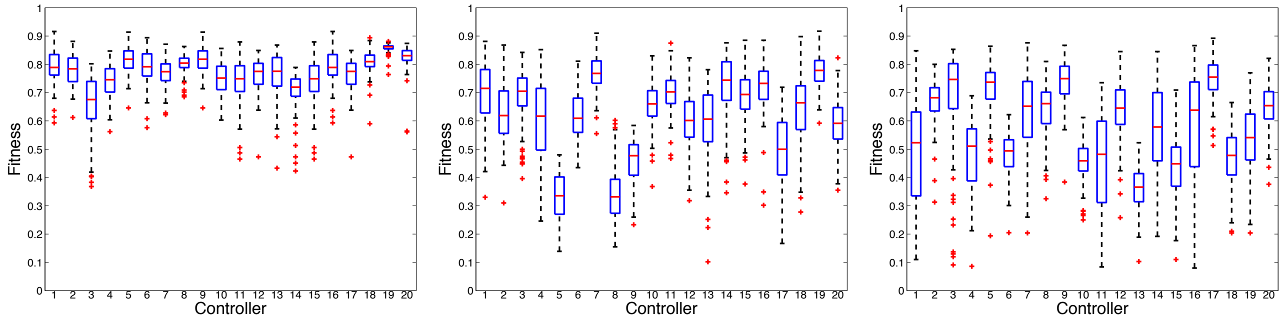
The difference in compactness of the group can be appreciated in the selected trajectories shown in Fig. 7, taken from the controller with highest median for each learning approach. In the case of the homogeneous controller, robots follow an almost perfect line. The trajectories followed by the heterogeneous controller learned with individual movement factor show that robots tend to spread, while those learned with group movement factor are more compact. These trajectories reflect the overall behaviors obtained by most of the 20 solutions of each learning approach.

In order to validate the results obtained in simulation, we select the controller with highest median for each learning approach and test it on real robots. We run 20 experiments for each solution. The initial positions and number of robots are the same as used for learning in simulation, but the evaluation time is reduced to 10 s in order to be able to keep track of the robots' positions during the whole evaluation due to the limited field of view of the overhead camera. Following the same scheme adopted in simulation, the local fitness function is computed on each robot using only its on-board resources, while the global fitness is computed externally



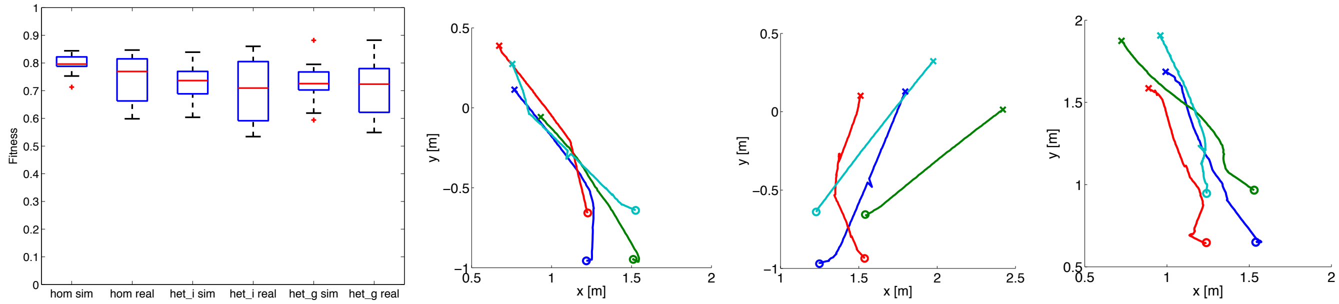
(a) (b) (c)

Figure 5: (a) Learning progress measured using the global metric for global homogeneous (centralized) learning . (b) Learning progress measured using the global (blue) and local (red) metrics for local heterogeneous (distributed) learning with individual movement factor. (c) Learning progress measured using the global (blue) and local (red) metrics for local heterogeneous (distributed) learning with group movement factor. The curves show the average of the 20 runs, and the error bars represent the standard deviation.



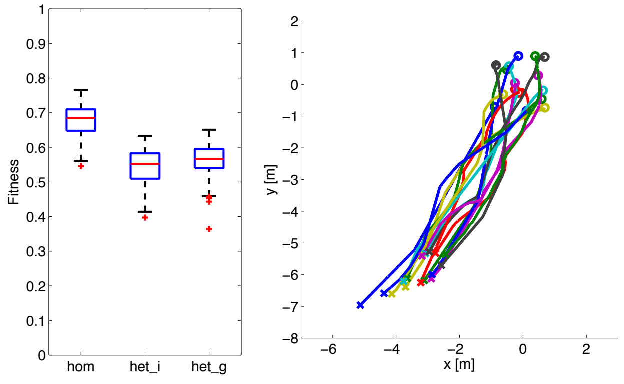
(a) (b) (c)

Figure 6: Performance measured with the global metric in simulation for the best solution found in each of the 20 independent learning runs with (a) global homogeneous (centralized) learning, (b) local heterogeneous (distributed) learning using individual movement factor and (c) local heterogeneous (distributed) learning using group movement factor. The box represents the upper and lower quartiles, the line across the middle marks the median, and the crosses show outliers for 100 evaluations of each controller.



(a) (b) (c) (d)

Figure 8: Evaluation with real robots for experiments of 10 *s* for the selected controllers. (a) Performance measured for 20 evaluations per controller with the global metric with real robots and in simulation for comparison of the selected controllers for: global homogeneous learning (*hom real* and *hom sim*), local heterogeneous learning using individual movement factor (*het_i real* and *het_i sim*), and local heterogeneous learning using group movement factor (*het_g real* and *het_g sim*). Trajectories of a single experiment with real robots for: homogeneous (b), heterogeneous with individual movement factor (c), and heterogeneous with group movement factor (d). The initial positions are marked with a circle, the final positions are marked with a cross.



(a) (b)

Figure 9: Evaluation in simulation using 16 robots for experiments of 60 *s*. (a) Performance measured for 100 evaluations per controller using the global metric for: homogeneous learning (*hom*), heterogeneous learning using individual movement factor (*het_i*), and local heterogeneous learning using group movement factor (*het_g*). (b) Trajectories of a single experiment of homogeneous learning.

given the information provided by the overhead camera.

Fig. 8a shows the performance for the best controller from each learning approach both in simulation and reality. The shorter duration of experiments (10 *s* as opposed to 60 *s*) implies that the initial stage of aggregation and alignment of the robots represents a larger fraction of the total time, and therefore the performances for the 10 *s* runs are lower than for the 60 *s* runs. Both in simulation and reality the homogeneous controller achieves the highest median, but there is a higher variance in the results with real robots that suggests that some modeling details are missing in the simulation.

The trajectories observed on real robots in Fig. 8 show the same differences in compactness and spreading for the three approaches that were previously seen in simulation.

The last set of experiments that we conducted, in this case in simulation only, consisted in increasing the number of robots from four to 16 to see how the different controllers learned with four robots generalize to larger group numbers. Robots were initially positioned with a uniform random distribution on a squared area of 2 *m* side, and random orientations. Fig. 9a shows performance measured with the global metric in simulation.

As expected, the homogeneous controller generalizes quite well. Robots form one line or various close lines and move straight in a very compact group (see Fig. 9b). We did not have the same expectations for the heterogeneous controllers, given that robots might assume specialized roles. In fact, the behaviors observed were similar to the four robot case, but the robots tend to spread, especially in the case of the controller learned with individual movement factor. The compactness factor is also lower due to the fact that it is impossible for each robot to keep the same desired distance to every other robot in the group of 16, which results in lower performance values overall.

The resulting flocking behaviors can be better understood by looking at the video provided as supplementary material[1]. It shows experiments in simulation and with real robots for the three selected controllers and different number of robots.

Conclusion

We have seen that the fitness metrics used, based on Reynolds' rules, reflect an appropriate flocking behavior. They allowed us to obtain better and more robust solutions than in our previous work (Di Mario et al., 2014b) for both the centralized and the distributed learning. The use of the alignment factor helps to maintain the cohesiveness of the group, and also to match the local metric with individual movement factor with the global one. Additionally, we have shown that the learned controllers can generalize to groups of increasing number of robots, resulting in specially robust controllers in the homogeneous solution.

Our results show that the controllers learned in the homogeneous centralized approach have higher fitness and lower standard deviation than those learned in a distributed manner, although the centralized learning takes four times the evaluation time of the distributed strategies. Nevertheless, the best solutions found for centralized and distributed learning performed similarly, in simulation and in the different experiments with real robots.

In the case of the distributed learning with individual movement factor, group compactness and cohesiveness were reduced due to the prevalence of the individual movement while learning a collective task. Our solution to this issue was to define a new local metric with global movement factor, which mimics perfectly the global metric, yet it makes the distributed learning harder since four different particles evaluated together return very similar local fitness regardless of the difference in behaviors of the individual controllers (a typical credit assignment problem in distributed learning). Learning with homogeneous controllers using the local metric with group movement factor could be a way to bypass the credit assignment problem while still allowing for on-board learning with simple noisy sensors without the need of external hardware, but it would not decrease the evaluation time per iteration through parallel evaluations.

As continuation of this work, we will explore new strategies for heterogeneous distributed learning that might result in better and more consistent performances of collective tasks. We will explore other neighboring topologies of the PSO algorithm, as well as different ways of distributing the particles among the robots. In addition, we would like to explore learning in the presence of obstacles in order to generate obstacle avoidance at the group level.

[1] http://disalw3.epfl.ch/research/distributed_adaptation/ecal.mp4

Acknowledgements

This research was supported by the Swiss National Science Foundation through the National Centre of Competence in Research Robotics.

References

Antonelli, G., Arrichiello, F., and Chiaverini, S. (2008). Flocking for multi-robot systems via the null-space-based behavioral control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1409–1414.

Balch, T. and Arkin, R. (1998). Behavior-based formation control for multi-robot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939.

Baldassarre, G., Trianni, V., Bonani, M., Mondada, F., Dorigo, M., and Nolfi, S. (2007). Self-organized coordinated motion in groups of physically connected robots. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics*, 37(1):224–39.

Celikkanat, H. (2008). Optimization of self-organized flocking of a robot swarm via evolutionary strategies. In *International Symposium on Computer and Information Sciences*.

Chang, Y.-H., Chen, C.-L., Chan, W.-S., Lin, H.-W., and Chang, C.-W. (2013). Fuzzy formation control and collision avoidance for multiagent systems. *Mathematical Problems in Engineering*, volume 2013.

Di Mario, E. and Martinoli, A. (2014a). Distributed particle swarm optimization for limited time adaptation in autonomous robots. In *Eleventh Int. Symp. on Distributed Autonomous Robotic Systems, Springer Tracts in Advanced Robotics*, volume 104, pages 383–396.

Di Mario, E. and Martinoli, A. (2014b). Distributed particle swarm optimization for limited time adaptation with real robots. *Robotica*, 32(02):193–208.

Di Mario, E., Navarro, I., and Martinoli, A. (2014a). Analysis of fitness noise in particle swarm optimization: From robotic learning to benchmark functions. In *IEEE Congress on Evolutionary Computation*, pages 2785–2792.

Di Mario, E., Navarro, I., and Martinoli, A. (2014b). Distributed learning of cooperative robotic behaviors using particle swarm optimization. In *14th International Symposium on Experimental Robotics*, to appear in *Springer Tracts in Advanced Robotics*.

Etemadi, S., Vatankhah, R., Alasty, A., Vossoughi, G., and Boroushaki, M. (2012). Leader connectivity management and flocking velocity optimization using the particle swarm optimization method. *Scientia Iranica*, 19(5):1251 – 1257.

Floreano, D. and Mondada, F. (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(3):396–407.

Gauci, M., Chen, J., Dodd, T., and Groß, R. (2014). Evolving aggregation behaviors in multi-robot systems with binary sensors. In *Eleventh Int. Symp. on Distributed Autonomous Robotic Systems, Springer Tracts in Advanced Robotics*, volume 104, pages 355–367.

Hayes, A. T. and Dormiani-Tabatabaei, P. (2002). Self-organized flocking with agent failure: Off-line optimization and demonstration with real robots. In *IEEE Int. Conf. on Robotics and Automation*, pages 3900–3905.

Jin, Y. and Branke, J. (2005). Evolutionary optimization in uncertain environments: A survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *IEEE International Conference on Neural Networks*, pages 1942 – 1948.

Lee, S.-M. and Myung, H. (2013). Particle swarm optimization-based distributed control scheme for flocking robots. In *Robot Intelligence Technology and Applications*, volume 208, pages 517–524.

Lochmatter, T., Roduit, P., Cianci, C., Correll, N., Jacot, J., and Martinoli, A. (2008). SwisTrack - a flexible open source tracking software for multi-agent systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4004–4010.

Matarić, M. (2001). Learning in behavior-based multi-robot systems: Policies, models, and other agents. *Cognitive Systems Research*, 2:81–93.

Michel, O. (2004). Webots: Professional mobile robot simulation. *Advanced Robotic Systems*, 1(1):39–42.

Morihiro, K., Isokawa, T., Nishimura, H., and Matsui, N. (2006). Emergence of flocking behavior based on reinforcement learning. In *Knowledge-Based Intelligent Information and Engineering Systems*, volume 4253, pages 699–706.

Navarro, I. and Matía, F. (2011). A framework for collective movement of mobile robots based on distributed decisions. *Robotics and Autonomous Systems*, 59(10):685–697.

Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51:401–420.

Parker, L. E. (1997). L-ALLIANCE : Task-oriented multi-robot learning in behavior-based systems. In *Advanced Robotics, Special Issue on Selected Papers from IROS'96*, pages 305–322.

Pugh, J. and Martinoli, A. (2009). Distributed scalable multi-robot learning using particle swarm optimization. *Swarm Intelligence*, 3(3):203–222.

Pugh, J., Raemy, X., Favre, C., Falconi, R., and Martinoli, A. (2009). A fast on-board relative positioning module for multi-robot systems. *Special issue on Mechatronics in Multi-Robot Systems, IEEE Trans. on Mechatronics*, 14(2):151–162.

Pugh, J., Zhang, Y., and Martinoli, A. (2005). Particle swarm optimization for unsupervised robotic learning. In *IEEE Swarm Intelligence Symposium*, pages 92–99.

Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34.

Zhang, Y., Antonsson, E., and Martinoli, A. (2008). Evolutionary engineering design synthesis of on-board traffic monitoring sensors. *Research in Engineering Design*, 19(2):113–125.

Unraveling the genotype-phenotype map of evolving digital organisms

Miguel A. Fortuna[1], Luis Zaman[2], Charles Ofria[2], and Andreas Wagner[1,3,4]

[1]Institute of Evolutionary Biology and Environmental Studies, University of Zurich
[2]Department of Biology, University of Washington
[3]BEACON Center for the Study of Evolution in Action, Michigan State University
[4]Institute of Evolutionary Biology and Environmental Studies, University of Zurich

Digital evolution is a form of evolutionary computation in which self-replicating computer programs—digital organisms—evolve within a user-defined computational environment. This experimental tool provides a unique framework to explore the structure of a genotype-phenotype map. In this map, the sequence of instructions constitutes the genome of digital organisms and defines its genotype. In addition to produce an offspring, a digital organism may be capable of computing one or more logic operations (tasks) by executing the instructions of its genome. We call the number and identity of the tasks it can perform as the organism's phenotype. The ability to preserve the phenotype under genetic mutations (robustness) leads to the existence of genotype networks (i.e., a continuous network of genotypes—in which two genotypes are connected if one can be converted into another by a single point genetic mutation—having all the same phenotype). Indeed, genotype networks are indispensable for evolutionary innovations because they allow the exploration of novel phenotypes while preserving the old ones. This ongoing project will show that only the rare and complex phenotypes allow substantial innovation, and even those are constrained to a few common, simple and robust novel phenotypes.

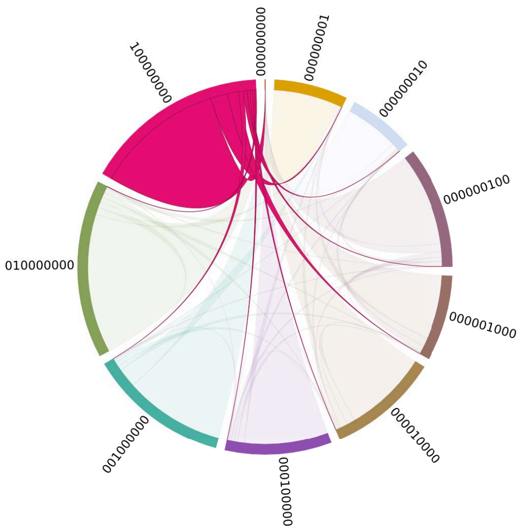


Figure 1: **Subset of the phenotype space.** Phenotypes are represented as a sequence of ones and zeros that indicate the logic function performed by the organisms with that phenotype (ordered clockwise following the complexity of the logic function performed, starting from the top). The arc length corresponding to each phenotype depicts the sum of the transition probabilities from that phenotype to the others (which are highly asymmetric). Transition probabilities of the most complex phenotype are highlighted. Only a few phenotypes are likely to be discovered from a given one, while most of them are unlikely to be encountered.

Incorporation of Emotions in the Orphibs' Agent Architecture

Nuno Barreto, Luís Macedo, Penousal Machado and Licínio Roque [1]

[1] CISUC
Department of Informatics Engineering
University of Coimbra
P-3004 516 Coimbra, Portugal
nbarreto@dei.uc.pt

Abstract

We describe the incorporation an existing emotion model, the Belief-Desire Theory of Emotion, into the architecture of the agents present in Orphibs II, a Life Simulation videogame prototype. To our knowledge, this represents the first time such a model is employed in video games.

Orphibs II

Orphibs is a 2D Life Simulation game (Barreto, et al. 2014) where the player takes the role of a caretaker, represented by a hand, for alien-like creatures called Orphibs. The game world, featuring a day/night cycle, is populated by several objects (food, toys and a bed).

Orphibs are aliens are driven by five needs: eat, fun, social interaction, reproduction and energy. The previous agent architecture was based on an extended Goal-Based Behavior (Barreto, et al. 2014) and featured genetically evolved personalities.

Extending the Orphib's Agent Architecture

The revised Orphib's agent architecture features 6 modules: (i) visual sensors, to gather the objects' states (ii) a memory mechanism that stores the last perceived object state (iii) an emotional model based on the Computational Belief-Desire Theory of Emotion (Reisenzein 2009) – CBDTE (iv) a component for executing actions (v) a database that stores desires and (vi) a reasoning mechanism which selects actions according to originated emotions.

One of the crucial points of the new architecture was the inclusion of beliefs: an object state that has a decaying probability of being accurate. Alongside the memory mechanism, Orphibs can confront memorized and perceived beliefs. This allowed the development of a new gameplay mechanic namely, information exchange during Orphibs' conversations.

The Emotional Model

Emotions are generated whenever beliefs and desires are fed into the comparators and comprise: an intensity; a type, which identifies the generated emotion; and a sign. Since more than one emotion can be generated from the comparators, the agent reacts to the most intense one.

Emotions play an important role in the Orphibs reasoning system as actions are chosen, at random, from the top three with the most positive emotions. Besides influencing the reasoning system, emotions are also expressed visually through facial expressions. Yet, this is only used to convey what the Orphib is feeling to the player, as illustrated in figure 1:

There are two moments when emotions can be generated: before an action is selected (pre-effect) and during its execution (post-effect). Pre-effect emotions can be described by fantasy emotions depicted in Reisenzein (2012). Since the Orphibs do not know the result of their actions, they express emotions resulting from the assumed outcome. Post-effect emotions, on the other hand, are the Orphib's reaction to events, more specifically, their reaction to newly perceived beliefs.

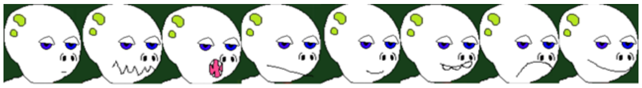


Figure 1 Orphib's facial expressions. From left to right: No Emotion, Fear, Surprise, Disappointment, Relief, Hope, Unhappiness and Happiness.

In the same manner as the genetically determined personalities present in Barreto, et al. (2014), we devised genetic emotional profiles by evolving the emotion's intensity functions (Reisenzein, 2009), as they are central to the reasoning mechanism.

Acknowledgements

This work was supported by the CISUC ICIS project CENTRO-07-0224-FEDER-002003 and the FCT Ph.D. Grant SFRH/BD/100080/2014.

References

Barreto, N., Macedo, L. and Roque, L. (2014). MultiAgent System Architecture in Orphibs II. In *14th International Conference on the Synthesis and Simulation of Living Systems*, pages 588-595, MIT Press, New York, NY.

Reisenzein, R. (2012). Extending the Computational Belief-Desire Theory of Emotions to Fantasy Emotions. In *11th International Conference on Cognitive Modeling*, pages 243-249.

Reisenzein, R. (2009). Emotional experience in the computational belief-desire theory of emotion. *Emotion Review*, 1(3):214-222.

Cooperative Coevolution of Morphologically Heterogeneous Robots

Jorge Gomes[1,2,3] and Pedro Mariano[3] and Anders Lyhne Christensen[1,2,4]

[1]BioMachines Lab, Lisbon, Portugal
[2]Instituto de Telecomunicações, Lisbon, Portugal
[3]Faculdade de Ciências, Universidade de Lisboa, BioISI, Portugal
[4]Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon, Portugal
jgomes@di.fc.ul.pt, plmariano@fc.ul.pt, anders.christensen@iscte.pt

Abstract

Morphologically heterogeneous multirobot teams have shown significant potential in many applications. While cooperative coevolutionary algorithms can be used for synthesising controllers for heterogeneous multirobot systems, they have been almost exclusively applied to morphologically homogeneous systems. In this paper, we investigate if and how cooperative coevolutionary algorithms can be used to evolve behavioural control for a morphologically heterogeneous multirobot system. Our experiments rely on a simulated task, where a ground robot with a simple sensor-actuator configuration must cooperate tightly with a more complex aerial robot to find and collect items in the environment. We first show how differences in the number and complexity of skills each robot has to learn can impair the effectiveness of cooperative coevolution. We then show how coevolution's effectiveness can be improved using incremental evolution or novelty-driven coevolution. Despite its limitations, we show that coevolution is a viable approach for synthesising control for morphologically heterogeneous systems.

Introduction

Cooperative coevolution (CCEA) has been advocated as a valuable approach for the evolution of heterogeneous multiagent systems (Potter et al., 2001). In the classic CCEA architecture (Potter and Jong, 2000), each agent evolves in an isolated population, and the individuals are evaluated by forming collaborations with individuals from the other populations. One key advantage of CCEAs is that since populations are isolated, it is possible for different populations to evolve radically different agents, with genomes of different lengths, and even to use different evolutionary algorithms. This vast heterogeneity has, however, rarely been exploited. Most previous works focus on the evolution of controllers for behaviourally heterogeneous, but morphologically homogeneous, multiagent systems (see for instance Potter et al., 2001; Yong and Miikkulainen, 2009; Nitschke et al., 2012). This means that all agents in the system have a similar complexity, similar sensor-effector capabilities, and use the same genotype representation.

Morphologically heterogeneous multirobot systems have shown significant potential in a number of applications (Howard et al., 2006; Dorigo et al., 2013; Duan and Liu, 2010). The cooperation between morphologically heterogeneous robots can, for instance, augment the capabilities of the group, allowing the achievement of tasks that are beyond the reach of a single type of robot. The behavioural control for morphologically heterogeneous systems is typically designed manually. This process can, however, be challenging, as behavioural control must integrate the capabilities of different robot types, in a way that the efficiency of the group becomes greater than if the different robot types worked independently without cooperation (Dorigo et al., 2013).

In this paper, we study if and how cooperative coevolution can be used to evolve effective controllers for agents with radically different capabilities in a task that requires tight cooperation. Cooperative coevolution is traditionally associated with a number of challenges (Wiegand, 2003) that stem from the intricate dynamics of coevolving two or more populations, where the evaluation of the individuals in one population depends on the individuals of the other populations. A key element in the evolution of cooperative behaviours is *synchronised learning* (Uchibe et al., 1998): populations should exhibit a mutual development of skills, in order to avoid loss of fitness gradients and convergence to mediocre stable states. We will study how the presence of radically different agents affects the mutual development of skills.

Our experiments are based on a simulated item collection task, where a ground robot with very limited capabilities, must cooperate with an aerial robot with a significantly more complex sensor-effector configuration. We explore different task variants to study how the differences between the agents, regarding the skills that must be evolved, affect the performance of cooperative coevolution. We then try to improve the effectiveness of coevolution with two techniques found in previous works: incremental evolution (Doncieux and Mouret, 2014) and novelty-driven cooperative coevolution (Gomes et al., 2014). We compare the advantages and drawbacks of each technique, and study how they can contribute to the effective coevolution of behaviours for multirobot systems with morphologically heterogeneous robots.

Related Work

Morphologically Heterogeneous Systems

Heterogeneous multirobot systems are characterised by the morphological and/or behavioural diversity of their constituent robots. Behavioural heterogeneity is commonly employed to allow behaviour specialisation within the group (Nitschke et al., 2012). In morphologically heterogeneous systems, on the other hand, agents have varied actuation and sensing capabilities, and collaborate to take advantage of the collective set of capabilities (Dorigo et al., 2013).

The Swarmanoid project (Dorigo et al., 2013) studied morphologically heterogeneous robotic swarms. Some of the tasks that were approached include: a gap crossing task, where ground robots receive instructions from aerial robots (Mathews et al., 2010); an indoor navigation task, where aerial robots aid ground robots in navigating through cluttered environments (Ducatelle et al., 2011); and a search-and-retrieval task in a 3-D environment, where aerial, ground, and climbing robots cooperate (Dorigo et al., 2013). Other works outside the Swarmanoid project have also shown the potential of cooperation between ground and aerial robots, especially in detection, search, and rescue tasks (Duan and Liu, 2010). Aerial robots can be used to assist ground robots, providing valuable information related to the environment (Lacroix and Le Besnerais, 2011).

Morphologically heterogeneous systems also encompass systems composed of robots of a similar nature (e.g., ground robots only). Heterogeneity can be used to reduce the cost of the group, by assigning different sensor/actuator capabilities to different robots. The robots then cooperate to take advantage of each other's capabilities. In (Parker et al., 2004), capable leader robots assist sensor-limited robots in navigating indoor environments. Howard et al. (2006) extend this approach to a task where few complex robots cooperate with a large number of inexpensive robots to map the environment and establish a sensor network. Grabowski et al. (2000) also study a mapping and exploration task, using two types of ground robots equipped with complementary sensors. Candea et al. (2001) study the coordination of different robot types in the context of the RoboCup competition.

Cooperative Coevolution

In the studies on multirobot systems discussed above, distributed control was achieved by manually designing the behavioural rules of the individual robots. Previous works have shown that this can be a challenging task, since the decomposition of the desired global behaviour into individual behavioural rules is often complex and inconspicuous (Dorigo et al., 2004). This challenge is exacerbated in heterogeneous systems (Dorigo et al., 2013), as behavioural control must be able to integrate the different abilities of different robot types to work in synergy towards the achievement of a common goal. One possible solution for this problem is the use of evolutionary algorithms to synthesise robot controllers (Dorigo et al., 2004; Uchibe et al., 1998; Potter et al., 2001). Besides automating the controller design, evolutionary algorithms have the potential to discover optimal solutions for the problem, and to discover diverse, unexpected solutions.

Cooperative coevolutionary algorithms are a natural fit for the evolution of heterogeneous multiagent systems (Potter et al., 2001). The classic cooperative coevolution architecture (Potter and Jong, 2000) operates with a system comprising two or more populations. Each agent is typically assigned to a separate population. The individuals of a population are evaluated by forming teams with individuals from the other populations. The fitness gradient is therefore relative: it is strictly a function of the individuals' contribution within the context of the other populations.

One advantage of CCEAs is that, due to the separation of populations, an arbitrary level of heterogeneity can be accommodated within the system. Cooperative coevolution, however, is typically applied only to morphologically homogeneous systems, focusing only on behavioural specialisation (Potter et al., 2001; Yong and Miikkulainen, 2009; Gomes et al., 2014; Nitschke et al., 2009, 2012). There have only been few reports of successful evolution of morphologically heterogeneous systems, and in the reported studies, agents had only minor morphological differences, for instance: a keepaway soccer task where agents have different moving and passing speeds (Gomes et al., 2014); a foraging task where agents have different movement speed and sensing ranges (Yang et al., 2012); and a predator-prey task where the predators have slightly different linear and turning speeds (Blumenthal and Parker, 2004).

Overcoming Coevolution Challenges

In a CCEA, the search space of each population is constrained by the individuals in the other populations. The search space is thus constantly changing, and the fitness of an individual can vary significantly depending on with which collaborators it is evaluated. This dynamic can cause two known pathologies: convergence to mediocre stable states (also known as relative *overgeneralisation*) and loss of fitness gradient (Wiegand, 2003). Convergence to mediocre stable states occurs when populations are unable to further improve their individuals, given the current set of collaborators drawn from the other populations. Previous works have shown that CCEAs tend to gravitate towards equilibrium states, not necessarily optimal solutions, which can impair their effectiveness (Panait, 2010). Loss of fitness gradient is more common in competitive coevolution, but can also appear in CCEAs: it occurs when a population reaches a state such that the other populations lose the fitness diversity necessary for meaningful progress (Wiegand, 2003).

We hypothesise that when populations have to evolve substantially different skills, with different complexity, a lack of synchronised learning is more likely to occur, causing convergence to stable states and/or loss of fitness diversity. In this paper, we study this effect and how related issues can be mitigated. To this end, we evaluate incremental evolution (Gomez and Miikkulainen, 1997) and novelty search (Lehman and Stanley, 2011; Gomes et al., 2014) as means to improve coevolution's effectiveness.

Incremental Evolution In an incremental evolution scheme, the goal-task is decomposed in simpler tasks for which solutions are easier to find (Gomez and Miikkulainen, 1997). A series of evolutionary stages are defined by the experimenter, and evolution moves from one stage to the next when the population reaches a sufficient performance level. Incremental evolution can be accomplished by defining a series of environments with increasing complexity (*environmental complexification*), or by defining a series of subobjectives (*staged evolution*) (Doncieux and Mouret, 2014). Incremental evolution has been used together with cooperative coevolution in a number of previous works (Nitschke et al., 2012; Yong and Miikkulainen, 2009; Uchibe and Asada, 2006). In our work, we evaluate incremental evolution, staged evolution in particular, as a way to encourage synchronised learning among the populations.

Novelty Search Novelty search is a widely recognised approach for overcoming premature convergence (Lehman and Stanley, 2011; Gomes et al., 2015). In recent work, Gomes et al. (2014) proposed a novelty-based method for avoiding convergence to equilibrium states in cooperative coevolution. The proposed technique (*NS-T*) relies on team-level behaviour characterisations, and rewards behaviourally novel collaborations in addition to high-fitness ones, as typically done in CCEAs. The team-level characterisations capture what the team as a whole achieves, without discriminating what each agent does for the team. It is shown that by rewarding individuals that cause novel collaborations, an evolutionary pressure towards novel equilibrium states is created. As there is a more effective exploration of the solution space, *NS-T* can reach collaborations associated with higher fitness scores more often than a traditional CCEA, and can evolve a diverse set of solutions for a given task in a single evolutionary run (Gomes et al., 2014).

Cooperative Item Collection Task

In the simulated task we use in this study, an aerial robot must assist a ground robot in collecting items randomly dispersed in an unbounded environment, see Figure 1. The ground robot has significantly fewer sensory capabilities than the aerial robot (detailed below). There is no direct communication between the two robots: they can only sense the relative position of each other when in close proximity. To accomplish the task, the aerial robot must learn to find the items and to guide the ground robot towards them. Complementary, the ground robot should follow the aerial robot and collect the items. The two robots must cooperate so that they do not lose track of one another.

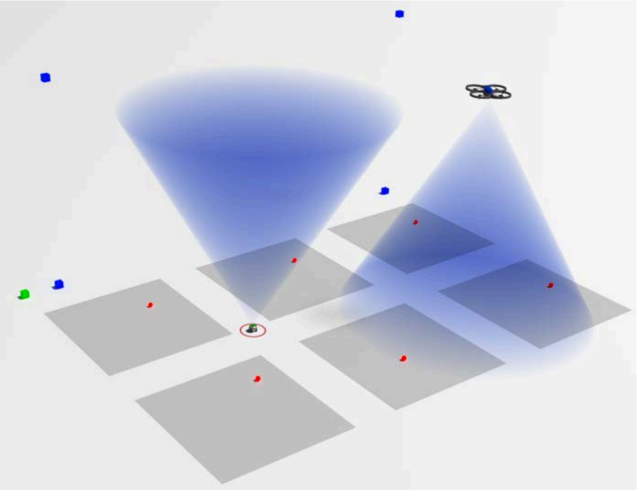


Figure 1: Cooperative item collection task. The red spheres are the items to be collected. One item is placed in each of the grey zones. The green cube indicates the starting position of the ground robot, and the blue cubes indicate the possible starting positions of the aerial robot. The blue cones depict the viewing range of the robots. The red circle around the ground robot depicts the range of its item sensor.

Robots Setup

Each robot is independently controlled by a neural network. The normalised sensor values are fed to the neural network, and the outputs control the actuators of the robot. We describe the sensor-actuator configuration of each robot below.

Ground Robot The ground robot has the ability to collect (remove) items from the environment. To collect an item, the robot simply has to pass over it. The ground robot is equipped with eight binary sensors:

- Four binary sensors to detect items within a range of 10 cm (depicted in Figure 1). Each sensor reads whether an item is present or not in the respective quadrant.
- Four binary sensors to detect the presence of the aerial robot. These sensors model an upwards-facing camera with a view angle of 60°. Each sensor indicates whether the aerial robot is present in the respective quadrant of the viewing cone. If the altitude of the aerial robot is over 250 cm, the ground robot's sensors are unable to detect it.

The two outputs of the neural controller control respectively the linear speed (within [0,3] cm/step) and the turning angle (within [-$\pi/5$,$\pi/5$] rad/step) of the robot.

Aerial Robot The aerial robot is equipped with a downwards-facing camera with a view angle of 60°, and with a maximum working altitude of 250 cm. The robot is also equipped with sensors for locating itself in the environment. It has a total of 15 sensors:

- Six real-valued sensors to detect items. Each sensor returns the distance to the closest item within the respective horizontal section.
- Six real-valued sensors to detect the ground robot. Each sensor returns the distance to the ground robot if it is present in the respective horizontal section, or the maximum value if it is not.
- Current altitude.
- Distance and relative angle to the centre of the arena. These sensors are used to allow the aerial robot to localise itself in the unbounded environment.

The four outputs of the neural controller dictate the movement of the robot, relative to the current robot's heading: (i) thrust in the left-right axis, (ii) in the forward-backwards axis, (iii) in the up-down axis, and (iv) rotation around its centre. The maximum speed of the robot is 20 cm/step in any direction (seven times faster than the ground robot), and the maximum rotation speed is $\pi/10$ rad/step. There is no altitude limit. When the robot has an altitude of zero (grounded), it can only move in the upwards direction.

Task Variants

We use a number of task variants in which different skills must be learnt by the aerial robot before it can assist the ground robot in solving the task. This approach allows us to gain insight on how differences in the learning speed of the agents affect the performance of cooperative coevolution.

In all task variants, six items are spread over an area of 550x350 cm^2, with each item placed randomly inside a zone of 150x150 cm^2, see Figure 1. The environment is unbounded, and the robots are thus free to roam away from the items and from each other. A simulation ends when all items are collected, or when 1000 time steps have elapsed. Each candidate solution (pair of ground robot and aerial robot controllers) is evaluated in five independent simulations. The ground robot always starts in the upper left corner of the arena, while the initial conditions of the aerial robot depend on the task variant, see Figure 1. The task variants are described below:

Fix-Tog **– Fixed altitude, start together** The aerial robot has no control over its altitude, but remains at the ideal sensing altitude (250 cm) throughout the whole simulation. The aerial and ground robots start together in the upper-left corner of the arena.

Fix-Sep **– Fixed altitude, start separate** The aerial robot maintains the maximum sensing altitude. The two robots start in opposite sides of the arena, from where they cannot sense each other. This means that the aerial robot must first learn to find the ground robot.

Var-Tog **– Variable altitude, start together** The aerial robot starts on the ground, and can freely move up and down. The aerial robot must thus learn to take-off, and control its altitude in order to optimise the sensing range. The two robots start together.

Var-Sep **– Variable altitude, start separate** The aerial robot starts on the ground, and the two robots start in opposite sides of the arena. The aerial robot must thus initially learn to control its altitude and find the ground robot.

Methods

Base Cooperative Coevolutionary Algorithm

All the evaluated methods are implemented over the same coevolution architecture. One population evolves the controller of the ground robot, while the other population evolves the controller for the aerial robot. Every generation, each population is evaluated in turn. To evaluate an individual from one population, a team is formed with one representative from the other population. The representative from a population is the individual that obtained the highest fitness score in the previous generation, or a random one in the first generation. Only the individual currently being evaluated receives the fitness score obtained by the team. The fitness score of a team, F_i, corresponds to the number of items that were successfully collected during the simulation trial.

The neural network individuals of each population are evolved by NEAT (Stanley and Miikkulainen, 2002), a state-of-the-art neuroevolution algorithm that evolves both the weights and topology of the networks, and has been extensively used in evolutionary robotics. We use the NEAT4J[1] implementation and most of the default parameter values: each population has a size of 150 individuals, the mutation probability is 25%, crossover probability is 20%, the probability of adding a connection is 5%, the probability of adding a neuron is 3%, and the target number of species is 5.

Incremental Evolution

We define a series of sub-goals that must be accomplished before reaching the ultimate objective of collecting items (F_i). Incremental evolution was configured specifically to bridge the gap between the number and complexity of skills each robot has to evolve, by encouraging the development of skills in the aerial robot and the cooperation between the two robots. We consider the following sub-goals:

1. Minimise the difference between the aerial robot's altitude (a_t) and the near-maximum sensing altitude (A, 240 cm) over the simulation trial (T time steps). The goal is achieved when 20% of the current individuals achieve a score of at least 0.9.

$$F_a = 1 - min\left(1, \sum_{t \in [1,T]} \frac{|a_t - A|}{T \cdot A}\right) \qquad (1)$$

[1] `http://neat4j.sourceforge.net`

2. Maximise the time robots spend within the sensing range of one another (t_w). The goal is achieved when 20% of the current individuals achieve a score of at least 0.7.

$$F_w = t_w/T \tag{2}$$

The configuration of incremental evolution depends on the task variant. Incremental evolution was not used in the *Fix-Tog* variant. For the *Fix-Sep* variant, we considered two stages: $F_w \rightarrow F_i$. For the *Var-Tog* and *Var-Sep* variants, we used three stages: $F_a \rightarrow F_w \rightarrow F_i$.

Non-cooperative Incremental Evolution

This approach is similar to the incremental evolution described above, with one key difference: the ground robot only starts evolving when the last stage (F_i, collecting items) is reached. During the previous stages, only the aerial robot evolves, while the ground robot remains still in its initial position. The rationale of this approach is to develop essential skills in the aerial robot before starting coevolution.

Novelty-driven Coevolution

Novelty-driven coevolution was implemented as described in (Gomes et al., 2014), using the *NS-T* technique, which computes the individual's novelty scores based on the behavioural novelty displayed by the team in which the individual participated. The team behaviour characterisation, used to compute the novelty of each team, is a vector of four real values normalised to [0,1]: (i) number of items caught; (ii) time robots spent within the sensing range of one another; (iii) average distance of one robot to the other; and (iv) average distance of each robot to the closest item. The novelty score of each individual is combined with its fitness score through a linear scalarisation that gives the same weight to the novelty score and to the fitness score.

As suggested in (Gomes et al., 2015), we use a value of k=15 (nearest neighbours) for the novelty score computation, and the archive is randomly composed: every generation, four random individuals are added to the archive.

Results

Base Cooperative Coevolutionary Algorithm

We begin by studying the performance of the base CCEA in the different task variants. The highest fitness scores achieved throughout evolution are depicted in Figure 2. Besides the four task variants, we also present a baseline where the aerial robot is not present — the ground robot alone evolves to collect the items. Each evolutionary treatment was repeated in 30 independent evolutionary runs.

The results show that there are clear performance differences in the four variants. The CCEA can consistently find good solutions for the *Fix-Tog* variant. In the other task variants, where the aerial robot needs to develop certain skills before being able to cooperate, the CCEA's performance is significantly affected ($p < 0.001$, Mann-Whitney test). Coevolution displayed the lowest performance in the *Var-Sep* variant ($p < 0.001$), where the aerial robot had to evolve the most complex behaviour. It is, however, possible to achieve solutions of similar quality for all task variants, as evidenced by the results in Figure 2 (right). This suggests that the large differences in performance are not necessarily explained by task difficulty, but rather by the ineffectiveness of the evolutionary process in reaching good solutions for some variants.

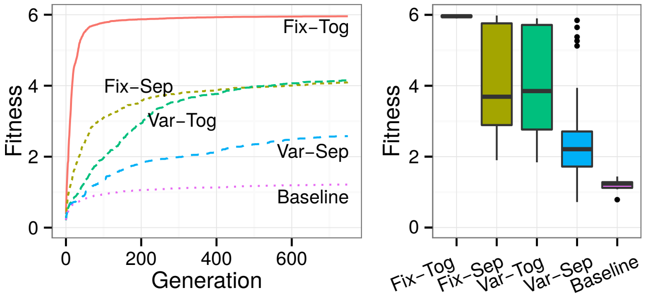


Figure 2: Fitness scores achieved by the basic *CCEA* in each of the task variants. Left: average highest fitness scores achieved at each generation. Right: boxplots of the highest scores achieved in each evolutionary run.

Table 1: N: number of successful / failed runs. Time within: average fraction of time the robots in the highest scoring solutions spent within the sensing range of each other.

	Successful runs		Failed runs	
Variant	N	Time within	N	Time within
Fix-Tog	30	96.5%	0	NA
Fix-Sep	12	70.6%	10	19.2%
Var-Tog	13	75.3%	11	14.4%
Var-Sep	5	45.1%	24	6.8%

To understand the reasons behind evolutionary failure, we looked at the highest scoring individuals evolved in each run. The evolutionary runs were divided into two sets: the successful runs, in which the highest fitness score was above 5; and the failed runs, in which the highest fitness score was below 3. We focused on the amount of time the robots spent within the sensing range of each other, which is directly related to the degree of cooperation between them, since the aerial robot cannot assist the ground robot if it is permanently outside its sensing range. Table 1 shows that in the successful runs, the robots stay within the sensing range of each other most of the time, across all task variants. In the failed runs, however, the scenario is different: the degree of cooperation between the robots is significantly lower ($p < 0.001$). Why do some of the evolutionary runs fail to evolve cooperation? We investigate two possible causes:

Loss of fitness gradients: Having agents that need to learn substantially different sets of skills can result in loss of fitness gradient in the populations (Wiegand, 2003). If one agent does not possess the skills necessary to make any impact in the performance of the team, fitness diversity might be lost. Consequently, the individuals cannot be adequately ranked, and evolution starts to drift. We investigated the loss of gradients by measuring the relative standard deviation (RSD) of fitness scores in each population, at every generation. A value close to zero means that fitness diversity is almost absent, indicating loss of fitness gradients. Higher values, on the other hand, indicate a rich fitness diversity.

Convergence to mediocre stable states: If the two populations fail to sustain a mutual development of skills, they can converge to a mediocre stable state (Panait, 2010): the individuals of one population can become over-adapted to the poor behaviours found in the other population. To analyse convergence to stable states, we resorted to a measure of team behaviour exploration, as done in previous works (Gomes et al., 2014). Team behaviour exploration is given by the mean behavioural difference between every two individuals evolved in a given evolutionary run, thus representing the dispersion of the individuals over the team behaviour space. Low values indicate that the individuals converged to a narrow region of the behaviour space, while higher values suggest that the behaviour space was reasonably explored.

The dispersion of fitness scores, shown in Table 2, suggest that the loss of fitness gradients is *not* the main cause for evolutionary failure. The average dispersion (RSD) is relatively high in both successful and failed runs, across all task variants and in both populations, meaning that fitness diversity is maintained. Regarding the behaviour space exploration, Table 3 shows that there is significantly less exploration in the failed runs than in the successful runs ($p < 0.001$). The relatively low degree of behaviour space exploration in the failed runs suggest that the main cause of evolutionary failure was convergence to mediocre stable states.

Improving Cooperative Coevolution

We evaluated three methods (described in the *Methods* section) to try to improve coevolution's effectiveness. The quality of solutions achieved with each method, for the three task variants where the base CCEA failed, are depicted in Figure 3. We also analysed the exploration of the behaviour space, using the measure described in the previous section. The results are shown in Table 4. Every evolutionary treatment was repeated in 30 independent evolutionary runs.

Incremental evolution (*Inc*) Incremental evolution was on average the highest performing approach, and significantly improved over the basic CCEA in all task variants ($p < 0.05$). The results in Figure 3 (left) show that incremental evolution tends to reach high fitness scores in fewer generations than the other methods. Incremental evolution initially rewards the robots for staying within sensing range of one another. As the robots are essentially forced to cooperate before reaching the final stage, evolution is less likely to get stuck in a mediocre stable state where the robots do not cooperate when collecting the items. Nevertheless, incremental evolution still fails in many evolutionary runs. The effectiveness of incremental evolution depends on the defined sub-goals, and as such it is possible that a substantially different configuration could yield better results.

Table 2: Average dispersion of fitness scores inside each population, at every generation. Dispersion is given by the relative standard deviation (RSD).

	Successful runs		Failed runs	
Variant	Ground	Aerial	Ground	Aerial
Fix-Tog	0.23	0.47	NA	NA
Fix-Sep	1.03	0.52	1.02	0.42
Var-Tog	0.47	0.70	0.96	0.52
Var-Sep	1.06	0.49	1.15	0.55

Table 3: Team behaviour exploration, given by the mean behavioural difference between every two individuals.

Variant	Successful runs	Failed runs
Fix-Tog	0.396±.07	NA
Fix-Sep	0.600±.06	0.305±.04
Var-Tog	0.619±.07	0.318±.07
Var-Sep	0.526±.07	0.272±.05

Incremental evolution is, however, associated with well known limitations (Doncieux and Mouret, 2014), as a great deal of domain knowledge is required to design effective evolutionary stages. We knew beforehand that maintaining an altitude close to 250 cm for the aerial robot and having the robots close to one another were desirable properties. If, however, we had little knowledge about the solution of the task, it would have been hard to shape the evolutionary process, with a risk of biasing it towards suboptimal solutions. Besides knowing which subgoals the evolutionary process must achieve, the experimenter must also specify what the order of the subgoals should be, and how to determine if a subgoal has been accomplished.

Non-cooperative incremental evolution (*NInc*) Non-cooperative incremental evolution displayed a relatively poor performance across all variants, and it was always significantly inferior to incremental evolution ($p < 0.05$), despite the similarities between these two methods. The rationale of this approach was to bootstrap the aerial robot before starting cooperative coevolution. With this approach, however, the aerial robot evolves a behaviour that is over-adapted to a static ground robot. When cooperative coevolution starts, and the ground robot starts moving, the aerial robot is not prepared to deal with it, and its previously learned behaviours tend to fail. This result supports previous works (Dorigo et al., 2013) that argue that when developing cooperative systems, the cooperation between the agents must be taken into account from the very beginning.

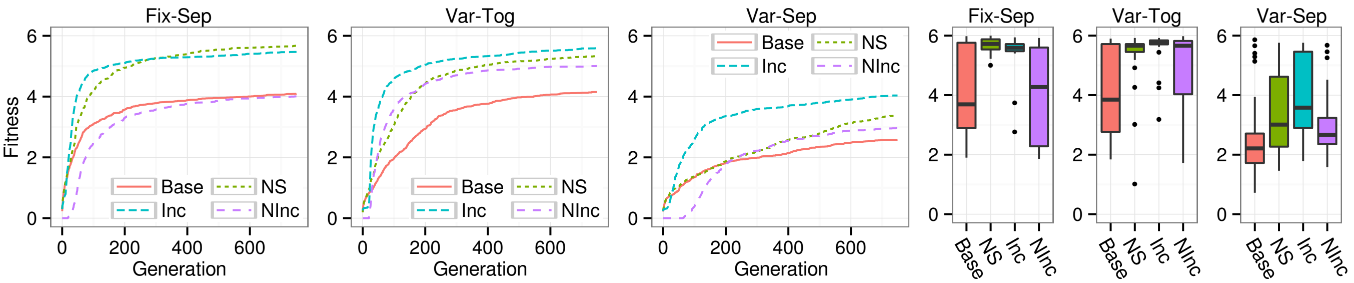


Figure 3: Fitness scores achieved in each task variant, with the base CCEA (*Base*), incremental evolution (*Inc*), non-cooperative incremental evolution (*NInc*), and novelty-driven coevolution (*NS*). Fitness corresponds to the number of items collected (F_i). Left: average highest fitness scores achieved at each generation. Right: boxplots of the highest scores achieved in each run.

Table 4: Average team behaviour exploration for the different evolutionary treatments and task variants.

Var.	Base	NS	Inc	NInc
Fix-Tog	0.40±.07	0.71±.03	NA	NA
Fix-Sep	0.44±.14	0.68±.05	0.54±.08	0.43±.14
Var-Tog	0.46±.15	0.68±.07	0.50±.07	0.47±.11
Var-Sep	0.31±.11	0.49±.07	0.43±.13	0.35±.09

Novelty-driven coevolution (*NS*) Novelty-driven coevolution avoids convergence to mediocre stable states in a completely different way than incremental evolution: it rewards the exploration of the behaviour space, without introducing biases towards specific behaviours. The results in Table 4 highlight this difference: novelty-driven coevolution exhibited a significantly higher degree of behaviour exploration than all other approaches in all task variants ($p < 0.05$).

Regarding the fitness scores achieved, *NS* significantly outperformed the basic CCEA in all variants ($p < 0.05$). The performance of *NS* is, however, inferior to incremental evolution in *Var-Tog* and *Var-Sep*. Novelty-driven coevolution was not always effective, as evidenced by the number of failed runs in the *Var-Tog* and *Var-Sep* variants (see Figure 3, right). In this item collection task, the team behaviour space can only be adequately explored if the two robots cooperate. If, for instance, the aerial robot does nothing at all, or simply flies away, the diversity of team behaviours that can be achieved is significantly compromised. Novelty-driven coevolution can get trapped exploring only a small region of the team behaviour space, and thus fails to discover high-quality solutions.

Conclusion

This work addressed the challenge of coevolving behaviours for cooperative multirobot systems where the robots have significant morphological differences. Our experiments relied on a task where a highly capable aerial robot must assist a relatively simple ground robot in collecting items. We used multiple task variants in which we varied the number of skills the aerial robot had to develop before being able to cooperate with the ground robot.

In the simplest task variant, where the aerial robot can immediately start cooperating with the ground robot, coevolution consistently found (near-)optimal solutions. Despite the disparity between the sensor-effector capabilities of the robots, and thus different complexity of their controllers, coevolution was able to sustain a mutual development of skills. In the task variants where the aerial robot had to develop additional skills before being able to cooperate, however, coevolution frequently failed. Our results showed that coevolution often converges to stable states where there is little or no cooperation between the robots.

We tried to improve coevolution's effectiveness using techniques described in previous works: incremental evolution; a non-cooperative version of incremental evolution, bootstrapping the more complex agent; and novelty-driven coevolution. Incremental evolution and novelty-driven coevolution significantly outperformed the basic CCEA, but they were still unable to consistently solve all task variants. Some evolutionary runs failed to evolve cooperation, especially when the aerial robot had to learn multiple skills before being able to cooperate. Novelty search was slightly less effective than incremental evolution, but it still managed to reach high-quality solutions, and relied less on the experimenter's knowledge. Our results also revealed that bootstrapping the more complex agent before the coevolutionary process is not effective, as it does not take in consideration the influence the agents have on the behaviours of one another.

Our experiments suggest that in order to coevolve agents with very different capabilities and sets of skills, the experimenter might need to modify the coevolutionary process to avoid mediocre stable states, and to encourage the evolution of the necessary skills for effective cooperation. This can be accomplished by incorporating domain knowledge into the process (incremental evolution) or by adopting a more open-ended evolutionary approach (novelty search). Despite this limitation, we showed that cooperative coevolution can yield good solutions, and can be considered a viable approach to evolve behaviours for morphologically heterogeneous multirobot systems. In future work, we are investigating if these results generalise to other tasks and robot types, and how cooperative coevolution might perform when different populations use different evolutionary algorithms.

Acknowledgements This research is supported by Fundação para a Ciência e Tecnologia (FCT), with grant SFRH/BD/89095/2012 and projects UID/EEA/50008/2013, UID/Multi/04046/2013, and EXPL/EEI-AUT/0329/2013.

References

Blumenthal, H. J. and Parker, G. B. (2004). Co-evolving team capture strategies for dissimilar robots. In *AAAI Artificial Multi-agent Learning Symposium*, volume 2. AAAI Press.

Candea, C., Hu, H., Iocchi, L., Nardi, D., and Piaggio, M. (2001). Coordination in multi-agent RoboCup teams. *Robotics and Autonomous Systems*, 36(2):67–86.

Doncieux, S. and Mouret, J.-B. (2014). Beyond black-box optimization: a review of selective pressures for evolutionary robotics. *Evolutionary Intelligence*, 7(2):71–93.

Dorigo, M., Floreano, D., Gambardella, L., Mondada, F., et al. (2013). Swarmanoid: A novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics & Automation Magazine*, 20(4):60–71.

Dorigo, M., Trianni, V., Şahin, E., Groß, R., Labella, T. H., Baldassarre, G., Nolfi, S., Deneubourg, J.-L., Mondada, F., Floreano, D., et al. (2004). Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots*, 17(2–3):223–245.

Duan, H. B. and Liu, S. Q. (2010). Unmanned air/ground vehicles heterogeneous cooperative techniques: Current status and prospects. *Science China Technological Sciences*, 53(5):1349–1355.

Ducatelle, F., Di Caro, G., Pinciroli, C., and Gambardella, L. (2011). Self-organized cooperation between robotic swarms. *Swarm Intelligence*, 5(2):73–96.

Gomes, J., Mariano, P., and Christensen, A. L. (2014). Avoiding convergence in cooperative coevolution with novelty search. In *International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, pages 1149–1156. IFAAMAS.

Gomes, J., Mariano, P., and Christensen, A. L. (2015). Devising effective novelty search algorithms: A comprehensive empirical study. In *Genetic and Evolutionary Computation Conference (GECCO)*. ACM Press. *In press.*

Gomez, F. and Miikkulainen, R. (1997). Incremental evolution of complex general behavior. *Adaptive Behavior*, 5(3-4):317–342.

Grabowski, R., Navarro-Serment, L. E., Paredis, C. J., and Khosla, P. K. (2000). Heterogeneous teams of modular robots for mapping and exploration. *Autonomous Robots*, 8(3):293–308.

Howard, A., Parker, L. E., and Sukhatme, G. S. (2006). Experiments with a large heterogeneous mobile robot team: Exploration, mapping, deployment and detection. *International Journal of Robotics Research*, 25(5-6):431–447.

Lacroix, S. and Le Besnerais, G. (2011). Issues in cooperative air/ground robotic systems. In *Robotics Research*, volume 66 of *Springer Tracts in Advanced Robotics*, pages 421–432. Springer.

Lehman, J. and Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2):189–223.

Mathews, N., Christensen, A. L., O'Grady, R., and Dorigo, M. (2010). Cooperation in a heterogeneous robot swarm through spatially targeted communication. In *Swarm Intelligence*, volume 6234 of *LNCS*, pages 400–407. Springer.

Nitschke, G. S., Schut, M. C., and Eiben, A. E. (2009). Collective neuro-evolution for evolving specialized sensor resolutions in a multi-rover task. *Evolutionary Intelligence*, 3(1):13–29.

Nitschke, G. S., Schut, M. C., and Eiben, A. E. (2012). Evolving behavioral specialization in robot teams to solve a collective construction task. *Swarm and Evolutionary Computation*, 2:25–38.

Panait, L. (2010). Theoretical convergence guarantees for cooperative coevolutionary algorithms. *Evolutionary Computation*, 18(4):581–615.

Parker, L., Kannan, B., Tang, F., and Bailey, M. (2004). Tightly-coupled navigation assistance in heterogeneous multi-robot teams. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1016–1022. IEEE Press.

Potter, M. A. and Jong, K. A. D. (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29.

Potter, M. A., Meeden, L. A., and Schultz, A. C. (2001). Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1337–1343. Morgan Kaufmann.

Stanley, K. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127.

Uchibe, E. and Asada, M. (2006). Incremental coevolution with competitive and cooperative tasks in a multirobot environment. *Proceedings of the IEEE*, 94(7):1412–1424.

Uchibe, E., Nakamura, M., and Asada, M. (1998). Co-evolution for cooperative behavior acquisition in a multiple mobile robot environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 425–430. IEEE Press.

Wiegand, R. P. (2003). *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, George Mason University.

Yang, J., Liu, Y., Wu, Z., and Yao, M. (2012). The evolution of cooperative behaviours in physically heterogeneous multi-robot systems. *Int. Journal of Advanced Robotic Systems*, 9(253).

Yong, C. H. and Miikkulainen, R. (2009). Coevolution of role-based cooperation in multiagent systems. *IEEE Transactions on Autonomous Mental Development*, 1(3):170–186.

Time Series Evolution for Integrating Developmental ~~Burglary~~ Processes

Sebastian von Mammen and Melanie Däschinger

Organic Computing, University of Augsburg, Germany
s.vonmammen@t-online.de
MelanieD@hotmail.de

Abstract

Material and virtual entities alike undergo developmental processes. If they replicate, they evolve and adapt to a variety of developmental processes. Based on the notion that these processes rarely constitute a direct translation from one state to the next, we have taken a step towards formally capturing evolutionary development of time series. We iteratively extend the Knapsack Problem to consider time, personal preferences, contradicting goals, external events, and changing environments and we show how the evolution of time series can be driven to address these various challenges. Within the interplay of genes and environment individuals shall develop who can survive optimally. The presented work constitutes a small step towards a more rigorous approach hinted at towards the end of the paper.

Introduction

When facing evolutionary challenges, organisms need to adapt—behaviourally within a lifetime, see for instance Noë and Laporte (2014), or genetically over the course of generations, as comprehensively investigated in the context of the evolution of the eye (Lamb, 2011). We consider adaptation in terms of properties of the organism, in terms of its anatomy or physiology. Yet, properties, just like environmental challenges, emerge one after the other and evolve over time (Gilbert and Burian, 2003). Accordingly, there is no fixed phenotype of an organism but a series of phenotypical states over time. This notion is supported by the nonlinear order of expression of genotypical information (Dang, 2014). In order to take a step towards this notion of organismal development, we have subjected a time series representation to evolutionary processes. Hereby, the transition to the next point in time of the organism's state depends on the previous one. At the same time, we assign such time-based individuals fitness values integrated over time, also considering changing environmental conditions.

In the remainder of this paper, we first reflect upon preceding approaches of evolutionary development. Afterwards, we present our concept of time series evolution (TSE) in the context of the knapsack problem. We extend the problem to a dynamic burglary process in terms of filling the backpack and in terms of multivariate external challenges. Step by step, we expand the representation of an individual and its fitness evaluation to arrive at a generic approach to time series evolution in dynamic environments. Finally, this approach is supposed to be used in complex systems, which consist of numerous interrelated processes, in order to find individuals who personate good and robust solutions. Experiments are presented to back up and to illustrate our rationale. We conclude our presentation with a summary and an outlook at ongoing and potential future work.

Related Work

There have been numerous computational approaches to condense the principles of growth and evolution. Stanley (2014) provide a valuable overview to this field. Abstract data structures have been automatically, interactively, immersively bred to take on a multiplicity of challenges (Sayama, 2014,Von Mammen and Jacob, 2009). For instance, the morphology of soft robots and their behaviour have been subjected to evolutionary algorithms (e.g. Rieffel et al., 2014), establishing accurate models of plant growth (e.g. Henke et al., 2014), and the design of computational hardware has been supported by evolutionary approaches (e.g. Bhattacharjee et al., 2015). Typically, such computational evolutionary developmental (or EvoDevo) approaches focus on a desirable end product. Yet, the product itself will be embedded in a context, it will be used, be worn, and vanish at some point in time. Similarly, its production does not entail a direct mapping from a concept to an artefact. Instead, it grows one step after the next—its existence changes over its lifetime (Gilbert and Burian, 2003). Hence, development is a proactive state of being rather than a phase with a well-defined beginning and end. Therefore, the field of adaptive computing approaches, including autonomic computing, which is tailored towards the continuous adaptation of computing systems (Wódczak, 2014) represents an important reference to the work presented in this paper. Even more so, do organic computing (Bernard et al., 2014) and morphogenetic engineering (Kowaliw et al., 2014) which target rather generic adaptive systems. However, the focus of this work still varies—it investigates the interplay of time series as generic developmental representations and genetic algorithms to allow their generational adaptation.

The 0-1-Knapsack Problem and TSE

The concept of time series evolution is motivated by the modelling challenges of developmental processes (Gilbert and Burian, 2003). Therefore, we identified the Knapsack Problem (KP) to be a theoretical problem which (a) considers development, which can (b) be expanded towards multi-objectivity, and (c) allows the introduction of the aspect of time.

The 0-1-Knapsack Problem can be defined as follows (Plateau and Nagih, 2010). n items of weights $w_1, ..., w_n$ and of values $v_1, ..., v_n$ are given, alongside a backpack with a weight limit of L. Valid solutions to solving this problem are vectors $a_1, ..., a_n$, with $a_i \in \{0, 1\}$ denoting whether or not the corresponding item is put inside the backpack and the constraint that the summed weight of all items does not exceed the limit, i.e. $\sum_{i=1}^{n} a_i w_i \leq L$. The best solution to the problem is the combination of items with the greatest overall value.

From 0-1-KP to Development

Now consider a burglar carrying the backpack during his raid. Thereby, the Knapsack Problem is modified from a static combinatorial planning challenge to a development process. It maintains the original goal of maximising the backpack's contained value but it allows for consideration of additional constraints. These can, for instance, be the burglar's varying physical condition, lock-picking challenges and other external factors.

We define an according *developmental series* as a vector $\vec{d} = (y_1, ..., y_n)^T$ which describes a number of decisions (encoded as numeric values) taken at the corresponding points in time $t \in \{1, ..., n\}$. Such developmental series represent solutions, or individuals in the context of evolutionary optimisation, given their sequences are valid. A function $g(t)$ may serve as generator of the developmental series, and for integrating the solution's state at an arbitrary point in time.

Heuristic 0-1-KP Solving

Before adding any further constraints, we briefly describe an example to show how a developmental series can approximate the optimal solution to the 0-1-Knapsack Problem. In particular, we consider the collectible items' value/mass-ratios and decide which one to pick up and which one to leave behind. In order to calculate a series' integrated value at a specific point in time, $g(t)$ would therefore be defined as follows (Eqn. 1). Figure 1 shows an evolved solution for a limit $L = 57$ for picking up a subset of 20 items that occur in descending order of their value/weight-ratios. Table 1 shows the given items.

$$g(t) = \begin{cases} g(t-1) + \frac{v_t}{w_t} & , if a_t = 1 \\ g(t-1) & , if a_t = 0 \end{cases} \tag{1}$$

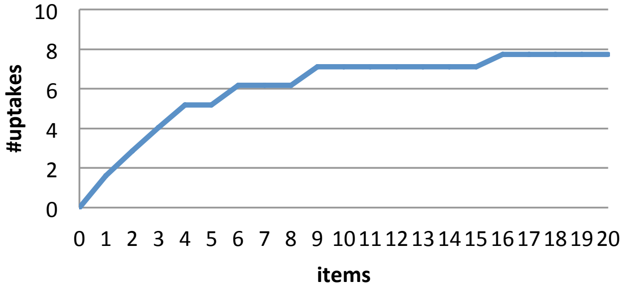


Figure 1: Uptake of items ordered according to their value/mass-ratios.

i	v	w	v/w	i	v	w	v/w
1	8	5	1.6	11	10	12	0.83
2	5	4	1.25	12	9	12	0.75
3	12	10	1.2	13	5	7	0.714
4	17	15	1.13	14	21	30	0.7
5	15	14	1.071	15	2	3	0.667
6	3	3	1.0	16	5	8	0.625
7	6	6	1.0	17	6	10	0.6
8	10	10	1.0	18	4	8	0.5
9	11	12	0.917	19	2	5	0.4
10	6	7	0.857	20	2	9	0.22

Table 1: Exemplary set of items for the 0-1-Knapsack Problem with indices i, values v and weights w.

Introducing Evolution

The generational evolution of developmental series can be realised by a Genetic Algorithm (Holland and Reitman, 1977). We need to assume that certain successions of states describe more successful developmental processes than others. Therefore, without loss of generality, we provide fitness values for each state of a developmental process, which results in a *fitness series* $\vec{f} = (z_1, ..., z_n)^T$ that defines the optimum. We rely on the mean quadratic error $\frac{1}{n} \sum^{n}{}_{i=1} (y_i - z_i)^2$ to derive a single fitness value for each developmental series. With an according set of genetic operators, the binary representation of the developmental series can be extended to arbitrary numeric values.

For the experiments presented throughout this paper, we deployed mutations with a chance of 10%. 30% of new offspring emerged from recombination of preceding specimen. We relied on fitness proportionate selection and elitism to keep the single best solution in the pool. Our tests regarding the population size included $20, 25, 40$, and 50 individuals.

Eventually, we stuck with 32, as it provided the best results. We let our experiments run for 40 generations, whereas the outcome typically converged after 10 generations.

Multi-objectivity

To give credit to the fact that a single optimisation criterion rarely captures the many facets of developmental processes, we extend an individual to a set of developmental series. Relating to the example above, the burglar might, for example, want to gather great value and achieve financial security fast. He might also want to maintain low weight of the backpack, especially for the first hours of the raid. In summary, we consider a population of individuals which consist of sets of developmental series DS. The elements $\vec{d_i} \in DS$ describe the development of some attribute i over n points in time. A function $g_i(t)$ can generate the respective values. As a result of the multi-series extension, the overall fitness of an individual can be calculated as the total of fitness values of all its development series over a given period of time. In analogy to $g_i(t)$, the fitness target values can be provided by functions $f_i(t)$. This generic representation not only allows for arbitrary fitness evolutions but may also serve for balancing the relative weights of the considered attributes.

Synthesis of Fitness Series

As can be expected, we observed that optimisation towards individual criteria, such as stalling weight growth for as long as possible, converges well. Loading up items from Table 1 with indices greater than 13, would, for instance, result in a mass development series $\vec{d}_{mass} = (0, ..., 30, 33, 33, 43, 43, 48, 57)^T$. Given the function in Equation 2, which rewards a late mass increase, yields a relatively high fitness value of 222.75.

$$f_{mass}(t) = 0.012 \cdot t^2 \cdot |\vec{d}_{mass}(t) - \vec{d}_{mass}(t-1)| \quad (2)$$

Adding the early financial security criterion f_{value} (Eqn. 3 and 4), which qualitatively inverts function f_{mass}' reward policy, would effectively work against this development. As a consequence, the given individual would perform very poorly in terms of the total value of 2.1 of considered fitnesses. Accordingly, the influence of several fitness series needs to be automatically balanced to reach a global optimum.

$$\Delta\vec{d}_{value} = |\vec{d}_{value}(t) - \vec{d}_{value}(t-1)| \quad (3)$$

$$f_{value}(t) = \begin{cases} (4 - 0.07 \cdot (t-1)^2) \cdot \Delta\vec{d}_{value}, if\ t < 9 \\ (0.05 \cdot \Delta\vec{d}_{value}, if\ t \geq 9 \end{cases} \quad (4)$$

Multiple Criteria & Diversity

In the following, we present results from applying multiple fitness criteria, including a fast increase in value (Eqn. 4), the maximisation of the value/weight ratio, and the slow accumulation of weight (Eqn. 2). We further considered the total weight and total value, and some inherited knowledge $f_{experience}$ that works as a generational memory for good decisions but is independent of any items' properties.

An individual optimised to address the 0-1-KP in accordance with Bellman's optimality equation (Montrucchio, 1986) would simply pick up the first seven items in Table 1. Yet, considering the given number of fitness series, such an optimised specimen would only achieve a final value of 66 and receive an overall fitness of 339.51 ($v/m = 82.55, v = 212.41, m = 16.55, experience = 14.0, other = 14.0$). Instead, the best GA-bred individual to address all the given factors achieved an overall fitness value of 380.95. Figure 2 shows its genotype and the relative fitnesses.

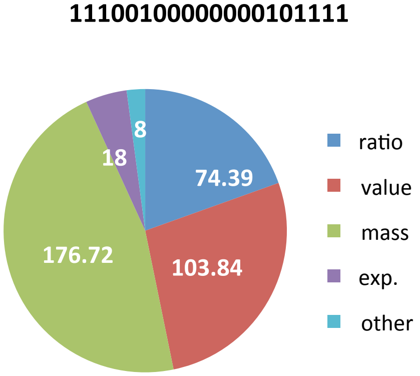


Figure 2: Top: The genotype of the best evolved specimen. Bottom: Its relative phenotypic fitness values.

We also found multiple specimen that achieved similarly high overall fitness ratings of about 300, relying on fundamentally different pick-up strategies. Figure 3 shows two rather diverse examples. Individual A achieves high overall fitness picking up items early on, whereas individual B focusses on the relative maximisation of value.

Dynamic Environments

So far, our model considers several developmental time series but only one set of ordered items. Although helpful for model development itself, the latter restriction is not adequate when considering complex developmental processes. Metaphorically speaking, the burglar may not be able to choose his raiding route upfront, not know what items to expect along the way in detail, nor would it be possible to plan in unforeseeable events, e.g. the appearance of a police patrol.

Based on these deliberations, one goal could be to find strategies that optimally fit a broad range of item spaces

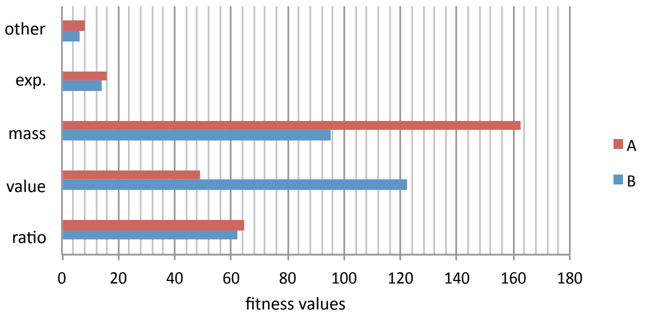


Figure 3: In this diagram, two individuals A and B are compared in terms of their relative fitness scores. Both individuals have achieved rather high overall scores of about 300.

(and orderings). Successful, fixed genotypes might reveal significant overlaps, in terms of picking up specific items, item properties or qualitative pick up sequences, that indicate preferable behaviours. When changing the order of the items, we discovered that the three items—originally indices 11, 13, and 19 from Table 1—were picked by the winners of two consecutively performed breeding experiments.

Dynamic Internal States

Another important modelling aspect is the condition of the burglar, or its internal state. Burglary, like everything else, requires energy. Therefore, without loss of generality, we assumed three breaks for snacks throughout the raid, with an overall decrease of recovery, see the blue columns in Figure 4. Their energetic maxima (at times $t = 2, 9, 17$) coincide with a tendency to pick up items. The minima (at times $t = 7, 14, 15, 20$), on the other hand, indicate exhaustion which nullifies the ability to pick up items. Energy is generally a rewarding dimension when considering process optimisation. For instance, it could be used to handle the aforementioned encounter with a police patrol—decreasing the specimen's supply at the time of the event (indicated by the red columns in Figure 4). Such strong constraints, of course, need to be considered during the evolutionary runs in order to provide valid solutions. The representation of external factors would encompass one n-dimensional vector $\vec{a}$ that quantifies the impact, a reference to the targeted developmental series $\vec{ts}$, and a set of constraints C that describe the relative impact of $\vec{a}$ on $\vec{ts}$

Dynamic System^2

Combining the concepts of dynamic system states and dynamically changing environments, we extend our model to consider DS^2, dynamic systems with a dynamical structure (Michel et al., 2009): Breaks cannot be accurately anticipated during a raid, police officers do not patrol neighbourhoods at regular times. Accordingly, in an experimental run comprising 25 generations, we offset the occurrence of police by 0 to 3 units and the occasion for breaks by 1 to 2 units at each other generation. The results can be seen in Table 2. As expected, the population adapts to the interval shifts. We recognise a corresponding cyclic pattern.

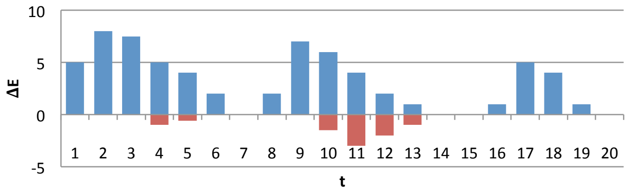


Figure 4: We define the internal state of the developing organism to depend on external events. In our example, a burglar replenishes his energy level three times throughout the raid (in blue). Encounters with a police patrol may cost energy (in red).

gen.	f	v	m	f_{max}-genotype	e/p
0-2	316	52	56	11101000000000001110	-
3-4	279	53	57	01110100010000001100	1/1
5-6	288	51	56	00111110000000000011	1/1
7-8	290	53	55	11101010000010000001	2/2
9-10	230	59	57	01101111010000100000	2/3
11-12	211	54	57	00101100110000110000	1/1
13-14	213	45	57	00001100001100100110	2/2
15-16	212	39	57	00000110001110100011	1/0
17-18	233	48	56	11000000110010111000	2/1
19-20	260	46	57	11000000011000111100	1/1
21-22	252	43	55	01100000000100111100	1/1
23-24	289	36	57	00100000000010011111	1/1
25-26	268	37	56	00001100000010001111	1/2

Table 2: Every other generation (gen.), a timing offset was introduced regarding energy intake (e) and policing (p) events. The best fitness values f_{max} indicate the adaptation of the population, the shifting pattern in item uptakes reflects the change of external events.

Summary & Future Work

In this paper, we have presented the concept of evolving time series in the context of developmental processes. Step by step, we extended the Knapsack Problem, turning it into a metaphorically understood burglar raid to suit the challenges faced in actual developmental processes. We started by merely evolving *developmental series*—in the given example a decision strategy for stealing specific items at particular points in time. Predetermined ideal progression was provided by *fitness series*, evolution of the developmental series ensured their approximation.

Next, partially contradicting desires by the decision maker were put to the test, necessitating prioritisation. Finally, we started considering the internal (physical) state of the burglar, introducing the notion of a generic time-dependent fitness criterion (energy) that is tightly interwoven with the environment. Our last experiments showed how time series evolution successfully adapts to dynamic environmental challenges.

Although it is already partially incorporated in the presented model, a major challenge is the accessible description and efficiently resolvable computation of constraints (a) among different developmental series and (b) across time steps. Depending on the resulting overhead, we hope to deploy our approach to biological developmental modelling and prediction. Currently, we are working on the architectural deployment of time series evolution as we can easily choose a manageable level of abstraction and since we expect actually applicable results.

References

Bernard, Y., Klejnowski, L., Bluhm, D., Hähner, J., and Müller-Schloer, C. (2014). Self-organisation and evolution for trust-adaptive grid computing agents. In *Evolution, Complexity and Artificial Life*, pages 209–224. Springer.

Bhattacharjee, D., Banerjee, A., and Chattopadhyay, A. (2015). Evodeb: Debugging evolving hardware designs. In *VLSI Design (VLSID), 2015 28th International Conference on*, pages 481–486. IEEE.

Dang, C. V. (2014). Gene regulation: Fine-tuned amplification in cells. In *Nature*, 511(7510):417–418.

Gilbert, S. F. and Burian, R. M. (2003). Development, evolution, and evolutionary developmental biology. In *Keywords and concepts in evolutionary developmental biology*, pages 61–8.

Henke, M., Huckemann, S., Kurth, W., and Sloboda, B. (2014). Reconstructing leaf growth based on non-destructive digitizing and low-parametric shape evolution for plant modelling over a growth cycle. In *SILVA FENNICA*, 48(2).

Holland, J. H. and Reitman, J. S. (1977). Cognitive systems based on adaptive algorithms. In *ACM SIGART Bulletin*, (63):49–49.

Kowaliw, T., Bredeche, N., Chevallier, S., and Doursat, R. (2014). Artificial neurogenesis: An introduction and selective review. In *Growing Adaptive Machines*, pages 1–60. Springer.

Lamb, T. D. (2011). Evolution of the eye. In *Scientific American*, 305(1):64–69.

Michel, O., Spicher, A., and Giavitto, J.-L. (2009). Rule-based programming for integrative biological modeling. In *Natural Computing*, 8(4):865–889.

Montrucchio, L. (1986). Optimal decisions over time and strange attractors: An analysis by the bellman principle. In *Mathematical Modelling*, 7(2):341–352.

Noë, R. and Laporte, M. (2014). Socio-spatial cognition in vervet monkeys. In *Animal cognition*, 17(3):597–607.

Plateau, G. and Nagih, A. (2010). 0–1 knapsack problems. In *Paradigms of Combinatorial Optimization, 2nd Edition*, pages 215–242.

Rieffel, J., Knox, D., Smith, S., and Trimmer, B. (2014). Growing and evolving soft robots. In *Artificial life*, 20(1):143–162.

Sayama, H. (2014). Guiding designs of self-organizing swarms: Interactive and automated approaches. In *Guided Self-Organization: Inception*, pages 365–387. Springer.

Stanley, K. O. (2014). Generative and developmental systems tutorial. In *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion*, pages 765–794. ACM.

Von Mammen, S. and Jacob, C. (2009). The evolution of swarm grammars-growing trees, crafting art, and bottom-up design. In *Computational Intelligence Magazine, IEEE*, 4(3):10–19.

Wódczak, M. (2014). Autonomic computing and networking. In *Autonomic Computing Enabled Cooperative Networked Design*, pages 3–16. Springer.

Heredity in Messy Chemistries

Nathaniel Virgo[1], Nicholas Guttenberg[2]

Earth-Life Science Institute (ELSI), Tokyo Institute of Technology
[1] nathanielvirgo@gmail.com [2] ngutten@gmail.com

Abstract

For natural selection to progress, there must be a sufficiently large evolutionary space to explore. In systems with template-based replication, this space is combinatorially large in the length of the information-carrying molecules. Previous work has shown that it is also possible for heredity to occur in much less structured chemistries; this opens the question of how the structure of a reaction network relates to the number of heritable states it can support, and in particular, how the number of heritable states scales with system size for a given network topology. Answering this question would allow us to map out the space of possible chemical mechanisms for heredity, and to identify places where they might be found in the space of organic chemistries that might have been found on the early Earth. We show that by linearising around a fixed point in a chemical reaction network and solving the corresponding eigenvalue problem, it is possible to detect the set of independent autocatalytic subnetworks that can operate in the vicinity of that point. We investigate an upper bound on the scaling of the number of such "autocatalytic cores" with the number of distinct chemical species, and show that the number of cores scales at best as $\log N$ in the case of unstructured networks, but that adding a strong energy constraint on the network topology allows it to scale linearly, which is the best possible case.

Introduction

How can natural selection emerge from chemistry? Out of the possible ways it can happen, which is easiest? These questions are important in the origins of life because an easier mechanism is a more parsimonious explanation for its historical occurrence on Earth, as well as a more probable route for it to emerge elsewhere.

There have been two major mechanisms proposed for heredity in the context of prebiotic systems. One, template-based replication, is familiar and well-studied from the point of view of modern organisms. The conditions for sustaining novel variation in template-based replicators have a solid theoretical basis (in the form of Eigen's error threshold), which can be used to understand the interplay between mutation and the ability to sustain novel variations.

Other proposed mechanisms have not yet received the same level of systematic treatment. In particular, several specific models have been proposed that exhibit compositional or attractor-based heredity (Segré et al., 1998; Szathmáry, 2000). Of particular note is the mechanism presented by Fernando and Rowe (2007) and refined using a different model by Vasas et al. (2012), in which heredity emerges from network autocatalysis. In these models, a chemical system has multiple attractors depending on which species are present, and rare events ("slow reactions") can add single molecules of a new species, thus shifting the system to a new attractor with a different composition and a different fitness.

We wish to know whether an evolutionary path can start out along the 'easy' route of composition-based heredity and then later evolve the more finely-tuned but higher-fidelity hereditary mechanisms we see in biology. As it stands, we have no systematic understanding of what determines or limits the degree of heredity in any given chemical system. Without this understanding, we lack the tools to evaluate whether such an evolutionary transition is feasible starting from a particular chemical system as a precursor. Even with the aid of numerical and laboratory experiments, this evaluation is limited by considerations of scale: there is a wide gap between the degree of evolutionary exploration that can be achieved in a lab-scale experiment versus the geological spatial and temporal scales involved in the origins of life.

To bridge this gap, we must understand how the degree of heredity — the number of heritable states — scales with the size of the system. If a system can support only a handful of heritable states and has no mechanism by which this number can increase, then very quickly it will explore all possibilities available to it and reach the limit of its evolution. On the other hand, a system in which the number of available heritable states scales with its size may be able to access a practically unlimited evolutionary search space. Even if a small version of such a system fails to discover a transition to unlimited heredity, a sufficiently scaled-up version may be able to do so.

Here we build upon previous work (Virgo and Ikegami, 2013; Virgo et al., 2014, Virgo et al., in press) by considering the dynamics of an arbitrary chemical system in the vicinity of a thermodynamically unstable fixed point; that is, one in which there is a concentration of 'food' that can be consumed but no species present that can consume it. We show that the dynamics near such a fixed point may be decomposed into the dynamics of sets of species that we call "cores," some of which are can be identified as autocatalytic cores in the sense of Vasas et al. (2012). This result comes from a simple application of the Perron-Frobenius theorem.

We then apply this approach to distributions of chemical networks with different structural constraints to investigate how the number of cores scales with the size of the network. This gives only an upper bound on the number of autocatalytic cores, and hence on the amount of heredity the system can support; but nevertheless our work represents the beginning of a research programme in which the potential for heredity is assessed through the development of a systematic theory, rather than the analysis of a few particular models.

Our recipe for identifying autocatalytic cores boils down to an eigenvalue problem to which Perron-Frobenius theory can be applied. This means it can be done in polynomial time, in spite of Andersen et al.'s (2012) result that finding all autocatalytic subnetworks of a reaction network is NP-complete. There is no contradiction here, since our algorithm solves an easier task: it seeks only first-order autocatalytic networks, and only those that feed directly on the food set, rather than on other species generated from it.

Chemical Reaction Networks

In this paper we use the formalism of *chemical reaction networks* along with mass action kinetics. This is a fairly standard way to specify the dynamics of chemical systems, as used informally by both chemists (see, e.g. Kondepudi and Prigogine, 1998) and mathematicians. A substantial amount of mathematical theory has been developed around the subject, starting with the papers of Feinberg (e.g. 1987); we will use only a little of this formalism, and we will present it informally through examples. For the more rigorously-mined, a good recent review is (Gunawardena, 2003).

The main assumption behind this type of model is that the chemistry takes place in a "well-mixed reactor," meaning that the concentrations don't vary over space. We will also assume that the concentrations are large enough that we do not need to consider stochastic effects, and that the species we care about are dilute enough that mass action kinetics are a good approximation.

A reaction network consists of a set of chemical species with their concentrations, together with a set of reactions between them. The reaction network represents the set of species that could in principle exist within the system, and the set of all reactions that would occur if the appropriate combination of reactants were present. Because concentrations may be zero, the set of species that are actually present and the reactions that actually occur may be very small subsets of the full network.

We consider the time-dependent vector of concentrations of chemical species $x_{\mathrm{A}}(t), x_{\mathrm{B}}(t)$, etc., with dynamics given by the standard assumption of mass action kinetics. Under this assumption a reaction $\mathrm{A}+2\,\mathrm{B} \longrightarrow 3\,\mathrm{C}$ proceeds at a rate $kx_{\mathrm{A}}x_{\mathrm{B}}^2$, where k is a rate constant assigned to this particular reaction. This reaction will consume A at a rate of $kx_{\mathrm{A}}x_{\mathrm{B}}^2$ moles per unit time, consume B at a rate of $2kx_{\mathrm{A}}x_{\mathrm{B}}^2$, and produce C at a rate of $3kx_{\mathrm{A}}x_{\mathrm{B}}^2$ moles per time unit.

In a regime where the free energy differences between species are small, every reaction is accompanied by a reverse reaction that converts the products back into the reactants, and thermodynamic principles constrain the ratio between the rates of the forward and backward reaction. This puts constraints on the dynamics of chemical reaction networks in the absence of a driving force: they must always eventually reach a single stable equilibrium state, and the approach to this equilibrium cannot be oscillatory. When the differences in free energy become large (the "irreversible regime") the reverse reactions occur at negligible rates, constraining reactions to flow only 'downhill' in the direction of decreasing free energy. We will make use of this below.

In addition, for any realistic chemistry, conservation laws will constrain the concentration vector to always lie within some subset of the space of possible values. For example, consider a network with reactions $\mathrm{A} \rightleftharpoons 2\,\mathrm{C}$ and $\mathrm{A} \rightleftharpoons \mathrm{B}$. No matter what the forward and reverse rates of these reactions are, neither of them can change the value of $x_{\mathrm{A}} + x_{\mathrm{B}} + 2x_{\mathrm{C}}$. Therefore, starting from an initial state $\mathbf{x}^0$, the dynamics are confined to the triangular region of concentration defined by the constraints $x_{\mathrm{A}} \geq 0$, $x_{\mathrm{B}} \geq 0$, $x_{\mathrm{C}} \geq 0$ and $x_{\mathrm{A}}+x_{\mathrm{B}}+2x_{\mathrm{C}} = x_{\mathrm{A}}^0+x_{\mathrm{B}}^0+2x_{\mathrm{C}}^0$. For a general network the dynamics are confined to a high-dimensional region known as the "invariant polyhedron".

However, there can also be fixed points on the boundary of the invariant polyhedron. Previous work (Virgo et al., 2014) showed that in the reversible regime these fixed points are always unstable and thus always lead to autocatalysis. In the next section we will concentrate on what can happen when linearising the dynamics around a boundary fixed point, whether in the reversible regime or not. We will show that such fixed points can be unstable along multiple different directions, corresponding to "autocatalytic cores," which can readily be identified from the network structure.

Autocatalytic Cores and Peripheries

Vasas et al. (2012) present their work in terms of the concept of an "autocatalytic core." We will show how the autocatalytic cores of a chemical reaction network can be identified given the structure of the network and its kinetic constants. Understanding the connection between network structure and the number of autocatalytic cores helps us to understand what properties a chemical system must have in order for it to exhibit limited heredity; and perhaps more importantly, it will help us to understand the circumstances under which this mechanism for "limited" heredity might not be so limited after all.

A key assumption in Vasas et al. (2012) is that the system is initially empty except for some "food set" of species, which is assumed always to be present at high concentration. Under such circumstances there will in general be some reactions that create other species directly from the food set; such species are called "food-generated." However, there may be other species whose concentrations remain at zero. There may be subsets of these non-food generated species for which adding a small concentration of any member of the set will cause the concentration of every member of the set to increase exponentially (or super-exponentially). These are the autocatalytic cores; they are sets of species which are autocatalytic, but which are also not food-generated, and hence they will be produced in the system only if they are seeded by some external process.

Vasas et al. (2012) propose a time-scale separation, whereby cores are "ignited" by slow reactions that have only a limited probability of occurring during the lifetime of a droplet; these slow reactions play a role analogous to rare mutations. We do not consider these slow reactions as part of the reaction network. In general an autocatalytic core will have side-products - species that are produced by reactions from the members of the core but whose presence will not ignite it. The set of all side-products, and all species that can be created from them, is called the core's "periphery."

Below we will assume that there are no food-generated species. This can be justified by saying that any reactions that can generate new species directly from the food set have already gone to completion, and the species thus generated have been incorporated into a new food set. In order for this situation to be interesting, we must also assume that some species' concentrations remain at zero.

These assumptions put us at a fixed point on the boundary of the invariant polyhedron of a reaction network, as defined in the previous section. We are on the boundary because some of the species' concentrations are zero, and we are at a fixed point because of our assumption that no reaction will proceed without the addition of some other species. Our question concerns what happens when small amounts of these species are added to the system.

The food set may consist of many species, but in general we will not care about distinguishing between them, and so in the chemical equations below we will use the symbol F to represent one or more members of the food set. Italic letters other than F will be used to represent sets of one or more species that are not members of the food set.

Our assumption that there are no food-generated species amounts to saying that the network contains no reactions of the form $F \longrightarrow X$ or $F \longrightarrow F + X$. Instead, all reactions must be of forms such as $X \longrightarrow Y$ or $F + X \longrightarrow F + Y$ that have at least one non-food species on the left-hand side. This guarantees that the state in which all non-food species have zero concentration is a fixed point.

We are interested in the behaviour of the system near this fixed point. The usual procedure for investigating dynamics near a fixed point is to write down the ordinary differential equation representing kinetics of the whole system, then do a first-order expansion around the fixed point to obtain an equation of the form $\dot{\mathbf{x}} = J\mathbf{x}$, where J is the Jacobian matrix. Below we give a recipe for constructing this Jacobian matrix directly from the network structure and the food concentrations.

Our first step is to note that in the vicinity of the fixed point, we only really care about the species with low concentrations, and not about the concentrations of the members of the food set. This is because each species contributes a term in the dynamics proportional to its concentration, so infinitesimal changes in the food set's concentrations have no appreciable effect on the dynamics of the near-zero species. Therefore we re-define the vector $\mathbf{x}$ so that its elements are the concentrations of the non-food species; the concentrations of the food species will be considered constant.

We then note that some reactions will not make any contribution to the linear approximation around the fixed point. A reaction such as $\mathrm{A} + \mathrm{B} \longrightarrow \mathrm{C}$ adds terms into the kinetic equations of the form $kx_\mathrm{A}x_\mathrm{B}$. This is second-order in the concentrations and hence will not appear in the Jacobian matrix. In the standard terminology of chemical kinetics, these are called second-order reactions. A reaction of the form $F + \mathrm{A} \longrightarrow \mathrm{B}$, however, produces a term of the form $kx_{\mathrm{F}_1} \dots x_{\mathrm{F}_n} x_\mathrm{B}$, which although nonlinear in the concentrations, is linear in the concentrations of non-food species. (Such reactions are termed "pseudo first-order.") Thus, the only reactions that need to be considered in forming the Jacobian are the first-order and pseudo first-order ones, i.e. the reactions that contain exactly one non-food species on the left-hand side.

As an example, let us suppose that reaction i has the form $\mathrm{F}_1 + \mathrm{F}_2 + \mathrm{A} \longrightarrow \mathrm{B} + 2\,\mathrm{C}$, where F_1 and F_2 are members of the food set. This reaction has only one non-food species, A, on the left-hand side. It will proceed at a rate $R_i = K_i x_\mathrm{A}$, where $K_i = k_i x_{\mathrm{F}_1} x_{\mathrm{F}_2}$ is assumed to be a constant in the regime we're considering. We therefore have that

$$\dot{x}_\mathrm{A} = dx_\mathrm{A}/dt = \cdots - K_i x_\mathrm{A} + \cdots ,$$

where the ellipses represent terms corresponding to other reactions. Similarly, we have that $\dot{x}_\mathrm{B} = \cdots + K_i x_\mathrm{A} + \cdots$ and $\dot{x}_\mathrm{C} = \cdots + 2K_i x_\mathrm{A} + \cdots$.

Differentiating by x_A, we see that reaction i adds a term $-K_i$ to the element of the Jacobian corresponding to $\partial\dot{x}_\mathrm{A}/\partial x_\mathrm{A}$, and terms K_i and $2K_i$ to the elements corresponding to $\partial\dot{x}_\mathrm{B}/\partial x_\mathrm{A}$ and $\partial\dot{x}_\mathrm{C}/\partial x_\mathrm{A}$.

This gives a general procedure for producing the Jacobian matrix around the fixed point corresponding to a given food set. First we identify all the reactions for which there is only one non-food reactant. Starting with a matrix whose entries are all zero, we iterate over every such reaction i. We calculate the value of K_i, given by k_i multiplied by the concentrations of any reactants that are members of the food set. We then subtract K_i from the diagonal element corresponding to the one non-food reactant, and add multiples of K_i, determined by the stoichiometric coefficients, to the elements corresponding to the partial derivative of the rate of change of the products with respect to the non-food reactant.

A matrix constructed by this procedure will be real, will not in general be symmetric, and will have negative elements only on the diagonal[1], with the other elements being either be positive or zero.

The dynamics near the fixed point depend on the eigenvalues and eigenvectors of the Jacobian. We know (by assumption) that this fixed point is stable along directions that correspond to changing only the concentrations of food species. However, it may be unstable along directions that correspond to adding small amounts of non-food species. Moreover, it can be unstable along multiple different directions, which correspond to adding different species to the system. In the remainder of this section we present a recipe for identifying these directions using Perron-Frobenius theory, and we then tie the resulting picture back to the notion of an autocatalytic core presented by Vasas et al. (2012).

Perron-Frobenius theory requires a matrix with only non-negative entries, but in fact this is not a problem. The linear approximation to the dynamics is given by $\frac{d\mathbf{x}}{dt} = J\mathbf{x}$, where J is the Jacobian matrix constructed as above. If we integrate this for a finite time period δt, we obtain $\mathbf{x}_t = e^{J\delta t}\mathbf{x}_0$. For small enough δt, the matrix $e^{J\delta t}$ has no negative values. (This can be seen by noting that it is approximated by $I + J\delta t$ for small δt, which also has no negative entries when δt is sufficiently small.) The eigenvectors of $e^{J\delta t}$ are the same as the eigenvectors of J, and its eigenvalues are given by $e^{\lambda_i \delta t}$, where λ_i is an eigenvalue of J. Thus, although J has negative entries on its diagonal, we can apply the Perron-Frobenius theorem to it as though it were a non-negative matrix of period 1.

We next state some fundamental results that follow from the application of the Perron-Frobenius theorem to this matrix. Formal proofs will be deferred to a future publication, but most follow immediately from the application of the Perron-Frobenius theorem, details of which may be found in any advanced linear algebra textbook, and an excellent overview is available online[2]. In doing this we will make much use of the directed graph corresponding to the matrix. In this graph, each node represents a species, and an edge is drawn from node A to node B if the corresponding (off-diagonal) element of J is non-zero, i.e. if a reaction takes one molecule of A as its sole non-food reactant, and has any amount of B as a product. It should be noted that this graph is a distinct entity from the reaction network itself; the graph encodes only partial information about the reactions, and only gives information about one fixed point. The great advantage in defining it is that a graph is a much simpler type of object to deal with than a reaction network. It should also be noted that the results below apply regardless of any thermodynamic considerations.

The first result is that if the matrix J is *irreducible*, meaning that the corresponding directed graph is strongly connected, then J will have a single eigenvector ν whose entries are all positive. The corresponding eigenvalue of J will be real, and its real part will be larger than the real part of all other eigenvalues of J. (This follows from the fact that the corresponding eigenvalue of $e^{J\delta t}$ has the largest absolute value.) If this 'leading' eigenvalue is negative then the fixed point is stable. (This can only happen in the irreversible regime, since this is a boundary fixed point). If the eigenvalue is zero then we need to go beyond the linear approximation to determine what happens, as discussed below.

The case of most interest is when the leading eigenvalue is positive, in which case the fixed point is unstable along the direction corresponding to ν. This eigenvector may be seen as a concentration profile; adding any amount of any species will cause all the concentrations to grow exponentially, while tending towards the same concentration profile ν, at least until they grow large enough that the linear approximation is no longer valid. Thus the system contains only one autocatalytic core, which consists of every single species in the system. This explains and generalises one of the key results of Virgo et al. (2014).

However, more generally the matrix J may be reducible. In this case the basic version of the Perron-Frobenius theorem does not apply. However, a reducible matrix may (by reordering of the rows and columns corresponding to the species) be written in "block upper-triangular" form, in which the blocks on the diagonal are irreducible. These irreducible blocks correspond to the strongly connected components of the digraph corresponding to J. We propose to refer to these as "cores," for reasons to be made clear below.

The eigenvalue spectrum of J is simply the union of the spectra of its cores. In particular, each core has its own dominant eigenvalue, which corresponds to an eigenvector in which all of the species that comprise the core have positive concentrations. A core's dominant eigenvalue may be positive, negative or zero; we refer to these as *autocatalytic cores*, *sub-catalytic cores* and *neutral cores*, respectively. We claim that this definition of autocatalytic cores corresponds to the concept proposed by Vasas et al. (2012). (But note that we are slightly modifying their terminology, in

[1] The diagonal elements may be positive, since a reaction might have the form $F + \mathrm{A} \to 2\,\mathrm{A} + X$ would add a positive term to the diagonal element of the Jacobian corresponding to x_{A}. Since our main interest is in how network autocatalysis emerges from the interaction between several species, we typically assume that such single-step autocatalysis reactions are absent from the network.

[2] http://en.wikipedia.org/wiki/Perron%E2%80%93Frobenius_theorem, accessed 16th March 2015.

that their term ‘core’ is simply short for autocatalytic core, whereas we allow cores that are not autocatalytic.) A core may have nodes that can be reached from it but from which there is no path back to the members of the core; in keeping with Vasas et al.’s terminology, we call these its periphery.

If a core is autocatalytic then adding any amount of any of its constituent species will cause all of them to increase in concentration exponentially. However, a subtlety arises because the cores are not necessarily independent of one another. If species A and B are members of different cores, there can be a link in the graph from A to B (i.e. a first-order reaction with A as a reactant and B a product), as long as there is not also a path from B back to A. If both cores are autocatalytic then adding a member of the core containing A will ignite *both* cores, whereas adding a member of B’s core will not ignite the core containing A.

One may thus draw a “dependency graph” between the cores; this is a directed acyclic graph (DAG), with a node corresponding to each core. This recovers the set of possible dependencies between autocatalytic cores that was identified in a less formal way by Vasas et al. (2012). Note that all cores participate in this dependency graph, not just autocatalytic cores.

There is another set of subtleties that must be covered, regarding the case of a neutral core. As in dynamical systems theory more generally, a neutral largest eigenvalue does not by itself tell us whether the fixed point is stable; instead it indicates that the linear terms have cancelled and nonlinear terms must be taken into account. Some of the consequences of this for reaction networks may be understood through a simple example: consider a network whose first-order reactions are $\mathrm{F} + \mathrm{A} \to \mathrm{B}$ and $\mathrm{B} \to A + \mathrm{P}$, where F is the only member of the food set, and P is an inert product. Constructing the Jacobian matrix for the non-food species A, B and P gives $\begin{pmatrix} -K_1 & K_1 & 0 \\ k_2 & -k_2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$, which has two cores, $\{\mathrm{A}, \mathrm{B}\}$ and $\{\mathrm{P}\}$, each with a leading eigenvalue of zero. If these are the only reactions then the fixed point really is neutrally stable, and we call $\{\mathrm{A}, \mathrm{B}\}$ a *catalytic core*, since the effect of adding A and/or B is to begin converting F into P without further affecting the total concentration of A and B.

However, if there are second-order reactions in this network then the situation may be different. For example, if there is a reaction $\mathrm{A} + \mathrm{B} + \mathrm{F} \to \mathrm{A} + 2\,\mathrm{B}$ then despite the inherent slowness of such a reaction (it is termolecular, and moreover depends on the product of the concentrations of two rare species) the concentrations will eventually be able to increase, moving away from the fixed point with “hyperbolic” (super-exponential) kinetics. On the other hand, if there is a reaction $\mathrm{A} + \mathrm{B} \to \mathrm{P}$ then the concentrations will decay sub-exponentially instead.

This completes our exposition of the linear dynamics of chemical systems near fixed points, and how the concepts of autocatalytic cores and their peripheries and dependencies can be identified from the topology and kinetic constants of the network. In the sections below, we will consider what properties the Jacobian matrix must have in order for a large number of different autocatalytic cores to be accessible from a given fixed point, and what structure might be required at the network level in order for a fixed point to have such properties.

It is worth first mentioning that there is a whole host of phenomena relevant for the origin of life and heredity that can only occur outside of the linear regime that we consider. The linear regime allows catalytic cores, which catalyse the conversion of food into other species, but it does not permit the consideration of catalysts that convert one non-food species into another, since the binding of a catalyst to its substrate is a second-order reaction. The systems that this excludes from our analysis include Eigen and Schuster’s (1979) concept of a hypercycle, as well as models based on Kauffman’s (1986) autocatalytic set model. It also excludes any system in which the recycling of nutrients is important, as well as systems containing “parasitic” cores that feed on previously ignited cores rather than the original food set. Nevertheless, we believe that a systematic investigation of the linear case is a good place to start in mapping out the space of possible chemical mechanisms for limited heredity.

When is limited heredity unlimited?

Up until how we have been concerned with the question of how many cores (and in particular, autocatalytic cores) a given network will have. We will now turn our attention to an important generalisation: given not a single network but a family of possible networks, how does the number of cores correlate with the size of the network?

This question is important because some forms of limited heredity are presumably less limited than others. A heredity mechanism that offers only a small, fixed number of attractors seems unlikely to lead to the evolution of sophisticated mechanisms like the genetic code that can allow the rate of information transmission to increase further. On the other hand, we might imagine a form of heredity that is “unlimited enough” to lead to the evolution of truly unlimited heredity.

Limited heredity is usually contrasted with the “unlimited” heredity provided by template replication (i.e. mechanisms resembling the replication of DNA), which can provide a number of heritable states that is exponential in the length of the string. For moderately long strings this can quickly become larger than the number of individuals that will ever be physically realised. We say the number of heritable states scales with 2^n, where n is the length of the strings; or simply that it scales with N, where N is the number of possible molecules.

It is easy to construct a contrived reaction network in which the number of autocatalytic cores scales with N. Trivially, we can let every reaction take the form $F + \mathrm{A} \to 2\,\mathrm{A}$, in which case every species is an autocatalytic core by itself and the number of autocatalytic cores is equal to N. Indeed, a simple model of template replication without mutation would also have this form.

In order to avoid putting into our models the very thing we wish to get out, we begin our investigation with an analysis of the number of cores (strongly connected components) that can be found in random directed graphs. That is to say, we imagine that a reaction network has been generated through some process, and that a linearisation has been taken around a boundary fixed point. We further imagine that the Jacobian matrix formed by linearising around this fixed point has no systematic pattern, and in particular has no correlation between the zero and non-zero entries.

We do not suppose that such an uncorrelated random matrix would arise from linearising around a fixed point of any actual reaction network. Rather, we make this assumption because it represents the least possible structure that could be put into any model. Given that we do not know what form the correlations are likely to take for real chemistries, it is the most parsimonious assumption available.

Under this assumption, we show numerically that the number of cores increases at most logarithmically with the number of species. (For a binary string chemistry, this would be equivalent to scaling linearly rather than exponentially with the strings' length.) Thus, any system that scales better than $\log N$ must have a Jacobian whose elements *are* systematically correlated in some specific way, and our next task is to show which kinds of correlations might give better scalings than $\log N$, and how they might arise without requiring the underlying reaction network to have an unreasonable amount of structure.

One possibility is that energetic considerations can help with this situation. Irreversible reactions must always flow downhill in terms of ΔG°, the difference in free energy between the products and the reactants. Reactions involving the food set can put energy into the system, allowing energetically uphill links in the directed graph. However, we show that if we limit the amount of energy that can be provided per food molecule then this constraint is enough for the number of cores to scale with N. As before, we arrive at this result by directly constructing a random matrix to be interpreted as the Jacobian, rather than by first generating a reaction network and then linearising around a fixed point.

It should be noted that our analysis in terms of random graphs does not distinguish between autocatalytic cores (the object of interest) and neutral or sub-catalytic cores. In the case of the energetic constraint we consider, one might expect most or all of the cores to be catalytic rather than autocatalytic, since when a molecule fissions there is no particular reason to expect both products to have a similar energy to the original molecule. Understanding the relationship between what is necessary to obtain an $O(N)$ upper bound while also causing a significant portion of the cores to be autocatalytic will be the focus of future work.

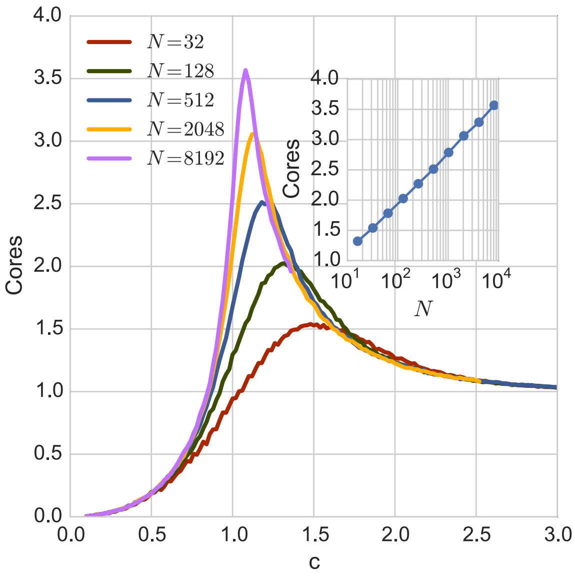


Figure 1: Distribution of cores for random directed graphs of varying sizes and link densities (c). The inset shows the scaling of the number of catalytic cores at the peak (critical) values of c, for different graph sizes. The number of catalytic cores increases logarithmically with the number of nodes in the case of critical random directed graphs.

Numerical Simulations

In order to test these ideas, we create randomly connected directed graphs, subject to constraints such as the energy differences in the reaction. Once we have generated a particular graph, we can extract the strongly connected components (SCC) of the graph using Tarjan's algorithm (Tarjan, 1972). We count the number of components of size greater than 1 (components of size 1 correspond to isolated nodes or compounds which do not participate in cycles). This tells us the number of catalytic cores in that particular reaction network.

The first case we consider is one in which we only specify the average number of links per node, c, but do not impose constraints on the directionality of reactions based on their ΔG°. This corresponds to the case of food molecules with a very large available energy from conversion. The resulting network is comparable to an Erdős-Rényi random graph (Erdős and Rényi, 1959), but is a directed graph rather than an undirected graph.

We generate many such networks of different sizes and compute the average number of catalytic cores as a function of network size (Fig. 1). Similar to the Erdős-Rényi case, we observe that there is a critical value of c which produces a maximum number of catalytic cores. However, even when considering graphs which are tuned to the optimum value of c, we find that the number of cores obtained scales as $O(\log N)$ in the number of nodes.

Next, we examine the case in which the food set has a

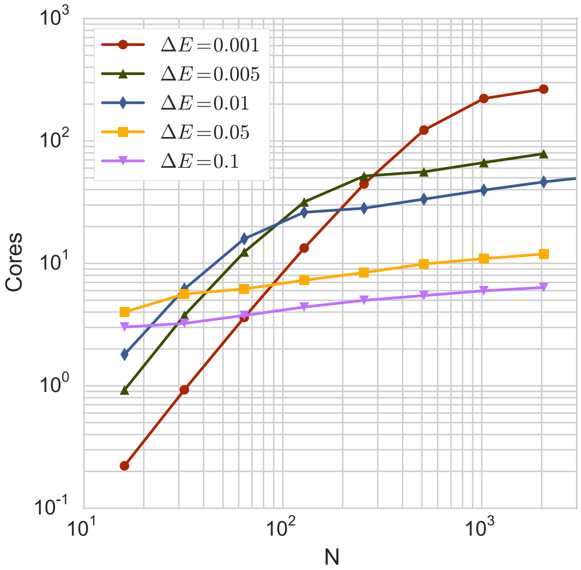


Figure 2: Scaling of the number of cores for graphs with uphill links limited to a change in energy ΔE. For each point, c is numerically tuned to maximise the number of cores.

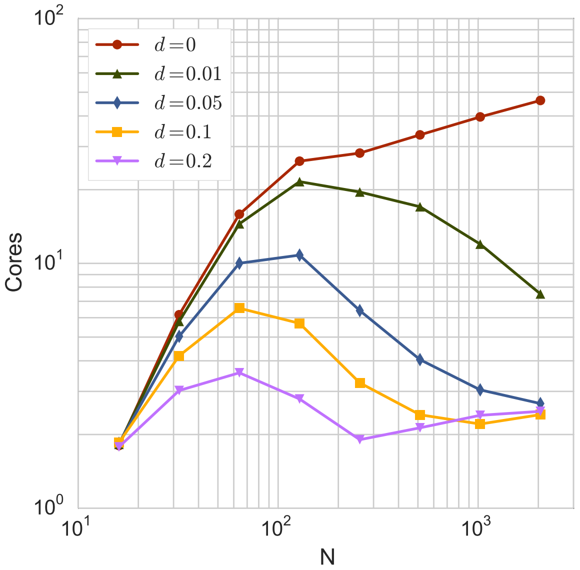


Figure 3: Scaling of the number of catalytic cores for graphs with a low-energy food set ($\Delta E = 0.01$), but with a small number of additional random links parameterized by d. As before, c is tuned to optimise the number of cores.

very low available energy compared to the energy range of the substrate molecules. We assign the substrate molecules random energies uniformly sampled in the range $[0, 1]$, and require that links can at most increase the energy of the product by a constant denoted ΔE. (Each link exists with constant probability Nc, unless it would violate this constraint, in which case the link is not added to the graph.) We find that as we make ΔE increasingly small, we can obtain a number of cores which scales as $O(N)$ in the number of nodes at the optimal c value (Fig. 2). However, at large N this scaling reverts to a slower scaling for finite ΔE or for non-optimal values of c. This may be because as the network grows, there is an increasing chance to find ways to chain together uphill reactions to create long loops in the energy space. It should be noted again that we have not shown whether these cores can be autocatalytic, since this would require a model of the full reaction network.

Finally, we consider what happens if we take the above system and add a high-energy food molecule that is very unreactive, in the sense that it reacts with only a few very specific molecules. This amounts to adding to the Jacobian a small number of non-zero terms that violate the energy constraint, connecting a node to one with a much greater energy; we implement this by simply adding completely random, uncorrelated links to the network with a fixed per-node probability d. If such a reaction connects between a pair of catalytic cores, then those two cores and all cores which connect between them via downstream links will merge and become a single core. This means that the more interconnected the downstream part of the DAG, the more sensitive it will be to disruption by structural 'noise' of this form. On the other hand it is possible for two cores not to be connected to each other even in a downstream direction. e.g. at the coarse-grained level, each core might be a completely isolated part of the graph. In this limit, a single random link can destroy at most a single core.

In the limit of a disconnected network, so long as the number of random links added scales more slowly than N, the overall coarse-grained structure can retain an $O(N)$ scaling. If the number of random links scales faster than N, it will dominate at large N and result in an approach to the random graph limit. When the number of additional links scales linearly with N (that is, each node has a certain fixed probability of having a reaction involving the high energy food set molecule), then the question is whether new cores are produced more quickly than they are destroyed as the graph increases in size. This means that there is a relationship between the chemical specificity (the average number d of reactions per node associated with that molecule) and the average core size s, such that if d is less than a quantity proportional to $1/s$ we can expect the $O(N)$ scaling to persist, whereas if it is greater then the graph will eventually be driven to the random graph limit.

In Fig. 3, we fix $\Delta E = 0.01$ and vary d. We find that for small graphs, this does not strongly influence the scaling of the number of catalytic cores, but as the graph size grows we observe an increasing separation into two distinct branches — one with the scaling of the structured graph, and one with the structure of the random graph.

Discussion

We have outlined a mathematical theory that determines how many distinct first-order autocatalytic cores can feed upon a given food set in a chemical reaction network. This theory may be stated in terms of the strongly connected components, or "cores" of a directed graph whose structure is determined by the network's topology; some of these cores may be autocatalytic cores in the sense of Vasas et al. (2012).This analysis gives us some insight about the kind of structure that must exist at the network level in order for the dynamics to be strongly dependent on the history of which species were added to the system, at least as long as the dynamics remain in the linear regime.

There are several reasons to be interested in such a question, but our main motivation in this work has been the mechanism for limited heredity proposed by Fernando and Rowe (2007) and Vasas et al. (2012). Those papers proposed specific but somewhat contrived artificial chemistry models that exhibited multiple autocatalytic cores; we have begun to answer the more general question of what structural properties a network must have in order to exhibit heredity through this mechanism. It is only by answering this question that we will be able to address the question of where limited heredity can be found in the space of organic chemistries that could have been instantiated on the early Earth.

However, we have also gone beyond the question of limited heredity by addressing the question of scaling. If the number of attractors in a system scales linearly with the number of possible molecules, then this "limited" heredity mechanism is just as unlimited as the heredity provided by template replication (i.e. DNA).

There are some strong limitations involved in studying only the linear behaviour around a fixed point. Perhaps the most severe of these is that our analysis does not allow "parasitic" cores that feed on previously ignited autocatalytic cores rather than the food set. A key point in both Fernando and Rowe (2007) and Vasas et al. (2012) is that a parasitic core may catalyse the production of the species on which it feeds, thus increasing rather than decreasing the fitness of its host. (See also Virgo et al. (2013) for an example where this occurs due to spatial patterning.) It would be useful to ask not only how the number of cores scales with the network size, but also how it scales with the number of cores that have already been ignited. Such questions would require an analysis of the full network rather than a single fixed point.

It is worth concluding our discussion with a brief mention of the Graded Autocatalysis Replication Domain (GARD) model (Segré et al., 1998; Markovitch and Lancet, 2012). On the face of it, this model offers a very different mechanism for limited heredity than that proposed by Vasas et al. However, GARD is at heart a linear model (with an additional nonlinear normalisation term), and the entries in the matrix are typically chosen to span several orders of magnitude, approximating the time scale separation between 'fast' and 'slow' reactions that Vasas et al. assume. Thus, it may be that an analysis similar to ours can be applied to GARD, showing that it exhibits limited heredity through the same mechanism after all. However, this is complicated by the fact that discrete, stochastic dynamics also seem to be important in GARD, and so we leave testing this hypothesis as a task for future work.

References

Andersen, J. L., Flamm, C., Merkle, D., and Stadler, P. (2012). Maximizing output and recognizing autocatalysis in chemical reaction networks is NP-complete. *Journal of Systems Chemistry*, 3(1):1–9.

Eigen, M. and Schuster, P. (1979). *The Hypercycle: A principle of natural self-organisation*. Springer.

Erdős, P. and Rényi, A. (1959). On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297.

Feinberg, M. (1987). Chemical reaction network strucutre and the stability of complex isothermal reactors I. the deficiency zero and deficiency one theorems. *Chemical Engineering Science*, 42(10):2229–2268.

Fernando, C. and Rowe, J. (2007). Natural selection in chemical evolution. *Journal of Theoretical Biology*, 247:152–167.

Gunawardena, J. (2003). Chemical reaction network theory for *in-silico* biologists. Lecture notes available at http://jeremy-gunawardena.com/papers/crnt.pdf.

Kauffman, S. (1986). Autocatalytic sets of proteins. *Journal of Theoretical Biology*, 119:1–24.

Kondepudi, D. and Prigogine, I. (1998). *Modern Thermodynamics: from heat engines to dissipative structures*. Wiley.

Markovitch, O. and Lancet, D. (2012). Excess mutual catalysis is required for effective evolvability. *Artificial life*, 18(3):243–266.

Segré, D., Lancet, D., Kedem, O., and Pilpel, Y. (1998). Graded autocatalysis replication domain (gard): kinetic analysis of self-replication in mutually catalytic sets. *Origins of Life and Evolution of the Biosphere*, 28(4-6):501–514.

Szathmáry, E. (2000). The evolution of replicators. *Philosophical Transactions of the Royal Society, Series B.*, 355:1669–1676.

Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160.

Vasas, V., Fernando, C., Santos, M., Kauffman, S., and Szathmáry, E. (2012). Evolution before genes. *Biology Direct*, 7(1).

Virgo, N., Froese, T., and Ikegami, T. (2013). The positive role of parasites in the origins of life. In *Artificial Life (ALIFE), 2013 IEEE Symposium on.*, pages 1–4. IEEE.

Virgo, N. and Ikegami, T. (2013). Autocatalysis before enzymes: The emergence of prebiotic chain reactions. In *Advances in Artificial Life, ECAL*. MIT Press.

Virgo, N., McGregor, S., and Ikegami, T. (2014). Self-organising autocatalysis. In Sayama, H. et al., editors, *Artificial Life 14*, pages 498–505. MIT Press.

Virgo, N., McGregor, S., and Ikegami, T. (In press). Complex autocatalysis in simple chemistries. *Artificial Life*.

Following Strategies Reduces Accidents, but Makes Outcomes Worse: Evidence from Simulated Treefrog Mating Scenarios

Giordano B. S. Ferreira and Matthias Scheutz

Department of Computer Science
Tufts University, Medford, MA 02155
{giordano.ferreira,matthias.scheutz}@tufts.edu

Abstract

Accidental matings happen in real environments where females end up with males they did not choose. In this paper, we investigate the frequency and changes in mated male fitness in accidental matings specifically in the context of the female choice of the gray treefrogs *hyla versicolor* based on the *best-of-n* and *minthresh strategy*, which are both hypothesized to be widely used in nature. Theoretical considerations as well as results from agent-based model simulations show how and why accidents occur and how the two strategies lead to different accident rates and reduced fitness values of the mated males.

Introduction

Mate choice is a biological selection process that is a critical determinant of the fitness of a species (Welch et al., 1998). Hence, much work in biology has focused on choice strategies, in particular, female choice strategies, and compared the utility of their outcomes, i.e., which strategy fares better based on the selection of the mate (Baugh and Ryan, 2009). However, little work has investigated the negative effects of such strategies when the chosen mate is not the one that ends up mating. This can happen, for example, when an impostor gets to mate instead of the chosen mate, or when an accidental mating occurs.

In this paper, we are particularly interested in investigating the negative outcomes caused by accidental matings in a biologically motivated mating task and compare two main female choice strategies from the literature with respect to the frequency of such accidents as well as the quality of the mates. Specifically, we will investigate how the fitness of male treefrogs changes as a results of accidental matings based on the *best-of-n strategy* (Janetos, 1980) vs. the *minthresh* strategy (Jennions and Petrie, 1997).

The rest of the paper is structured as follows. We start with background information about the task and introduce formal definitions of both strategies, followed by some facts about accidental matings for each strategy. Then we introduce the experimental setup, including the parameter space we investigated, followed by a presentation of the results together with an analysis showing the influence of the parameters on the number of accidental matings and also on the fitness of the mated males. Next we discuss when and why accidental matings can happen for each strategy, including the reason why using strategies reduces accidents. Finally, the conclusion summarizes our discoveries and proposes extensions for future work.

Background and Definitions

In previous work (Scheutz et al., 2010), we have investigated two main mate selection strategies using the gray treefrog *hyla versicolor* as an animal model in a biologically plausible mating task where female treefrogs located at the edges of a swamp have to choose a male mate from among a set of calling males situated in the swamp. The first strategy, called *best-of-closest-n* or "best-of-n" for short, requires females to select the best male within the *n* closest males. While how "best" is evaluated in female choice depends on the specifics of the species under investigation, in gray treefrogs females are attracted to the call quality of male callers (Gerhardt, 1994). In particular, the *pulse number* of a male call is a major determinant of the quality of a male treefrog, and this quality has, in fact, been linked to the fitness of the females' offspring (Welch et al., 1998).

The other strategy we have investigated in the past, *closest-above-minimum-threshold* or "min-threshold" for short, requires female treefrogs to select the closest male caller with a call pulse number greater than a *minimum quality threshold* θ. As with the first strategy, females listen to male callers in the swamp and then pick a caller based on the strategy's recommendation. With both strategies, the females sitting at the edges of the swamp will then start moving *in a straight line directly towards* the chosen male and when she arrives at the caller's location will mate with the male.

Next, we will make these notions formally precise. Let $D(f, m)$ denote the straight-line distance between a female f and male m treefrog in the swamp. Let *MALE* be the set of all males in the swamp and *FEMALE* the set of all females at the edges of the swamp, and let m_{pn} denote the pulse number of male $m \in$ *MALES*. Define the set of closest agents from a given set X to a given agent i as $c(i, X) = \{j \in X | \neg \exists k \in X[D(i, k) < D(i, j)]\}$ and we let $c^n(f, X)$ denote the set of the n closest agents from set X with respect to the location of female f.

Then we can define both strategies formally as in Scheutz et al. (2013):

- *best-of-n.* The selected male agent is $\underset{m \in c^n(f, MALE)}{\text{argmax}} (m_{pn})$ for the female f, i.e., the male with highest pulse number in the set of the closest n males.
- *minthresh.* The selected male agent is $\underset{m \in c(f, \{l \in MALE | l_{pn} \geq f_\tau\})}{\text{argmax}} (m_{pn})$, where f_θ is the minimum threshold of female agent f, i.e., the male with the highest pulse number above the minimum threshold among the closest males.

In our past work, we used an agent-based modeling and simulation environment to investigate various tradeoffs among those strategies, which allows us to both test behaviors observed in empirical experiments and formulate hypotheses for further evaluation in the real world. In Scheutz et al. (2010), for example, we used the simulation environment to determine whether one of the two strategies clearly *dominated* the other, i.e., whether there were parameter settings for *best-of-n* vs. *minthresh* such that one strategy consistently showed better average mated male quality than any of the others. The results from extensive simulations of larger parameter spaces showed that even though females using the *minthresh* strategy perform better for much of the parameter space compared to females using *best-of-n*, *minthresh* did not dominate *best-of-n* because there are regions of the explored parameter space where *best-of-n* performed better than *minthresh* for some parameter values.

In other work, we investigated how the two strategies would fare when males were re-positioning themselves in order to create calling sites that could increase their chances of being chosen by females (Scheutz et al., 2013). In this extended setting, males can either call remaining stationary in their chosen position or wander, leaping through the swamp to find a better location for calling that would improve their chance of mating. We hypothesized that staying near a high-quality male caller would increase the chance of mating for a male for two reasons. First, the high-quality caller will likely attract several females that will independently approach him, but once he mates with a female, he will stop calling. Thus, the males near him could become of interest to close-by females that were attracted to the location by the high-quality male. Second, when a female is leaping to the high-quality male, she might accidentally bump into another lower quality male that is close to the high-quality male, but directly in her approach trajectory. To simulate the wandering behavior, we allowed male frogs to use the same two strategies used by females, i.e., either *best-of-n* or *min-threshold*, to evaluate the quality and location of fellow male callers. The simulation results showed that mate quality overall improves when males are allowed to reposition themselves compared to non-repositioning males, and that this improvement was greater when females and male wanderers use the same strategy (Scheutz et al., 2013). However, it was unclear to what extent these differences were due to females in the area picking their second (or third, etc.) choices after the high-quality male already mated, and to what extent the fitness was actually lower than it could have been due to *accidental matings*, i.e., females bumping into males on their way to the chosen mate.

Specifically, we are now interested in determining the extent to which females mate by accident, i.e., the *frequency of accidental matings*, and the *average fitness of the accidentally mated males* compared to frequency of non-accidental matings and the average fitness for the chosen and mated males. These tradeoffs are not only important for understanding female choice in the context of treefrog matings, but also for evaluating the fitness of these strategies in general biological domains, at least for two reasons: (1) accidental matings could have very negative if not detrimental consequences for females and offspring, hence accidental mating frequency matters; yet, (2) lower quality mates, even though they might have negative consequences in the short term, might be able to preserve the variety in the gene pool in the long term and thus be overall positive for the species. We will, in the following, start with some general observations about accidental matings based on the definitions of the two strategies and then move towards agent-based simulations to be able to quantify tradeoffs that cannot be predicted based on general principles.

Facts about Accidental Matings

Start by defining an *accidental mating* as any mating that occurred involving a male that the female did not select based on her female choice strategy. Note that accidental matings can occur with both strategies when a female is moving towards a chosen male and ends up bumping into another non-chosen male while traversing the swamp.[1] However, accidental matings can also happen when none of the remaining males' call qualities meet the minimum threshold of the remaining females in the swamp using the *minthresh* strategy. For in that case, females will leave the swamp and might also bump into a lower-quality male by accident.

Definition Let m be a male frog and f be a female frog. We denote $F(m)$ as the fitness of the male frog m, and $D(f, m)$ as the distance between a female f and male m. Let d_{mate} be the *mating distance*. So, in order for a mating to occur, $D(f, m) < d_{mate}$.

[1] Note that treefrogs in that case will always mate, but that it is certainly possible to define a probability of accidentally mating in such cases and that this probability will then determine accidental mating frequency and male quality.

In the following, we will report a few facts about accidental matings. We will let

Fact 1 *Let n be the parameter for best-of-n strategy for all females in* FEMALE. *No female using* best-of-n *with $n = 1$ can mate by accident.*

Proof Suppose f is an accidentally mated female with $n = 1$, that m_{ac} is the male involved in the accidental mating, and that m_b the male chosen by f (both males in *MALES*). Then by definition, $D(f, m_{ac}) < d_{mate}$ and $D(f, m_b) > d_{mate}$. Therefore, $D(f, m_{ac}) < D(f, m_b)$. However, by definition of the *best-of-n* strategy for $n = 1$, the chosen male m_b is the closest male, contradicting $D(f, m_{ac}) < D(f, m_b)$.

Fact 2 *Let θ be the parameter for the* minthresh *strategy for females in* FEMALE. *Then no accidental mating can occur for males $m \in$* MALES *with $F(m) > \theta$.*

Proof If $F(m) > \theta$ for all $m \in$*MALES*, then by definition of *minthresh* each female using the strategy with θ will pick the closest male. Hence, there cannot be any male between the female and the chosen one (which would be closer).

Fact 3 *Let n be the parameter* for best-of-n strategy *for all females in* FEMALES. *For* best-of-n strategy*, accidental matings lower the average fitness of the mated males.*

Proof Consider the subset $M_f \subseteq$ *MALES* of all males that are the n closest to a given female f and let $m_b = argmax\{F(m)|m \in M_f\}$. Suppose a male m_{ac} accidentally mated with f. Then $D(m_{ac}, f) < m_b$, since females have a direct straight-line approach to males and thus $m_{ac} \in M_f$. By definition of *best-of-n*, $F(m_{ac}) < F(m_b)$. Therefore, the average mated pulses with accidents is strictly lower than without.

Fact 4 *Let n be the parameter for* best-of-n strategy *and let θ be the parameter for minthresh strategy. Furthermore, let $M_f \subseteq$* MALES *be the subset of all males that are the n closest to a given female $f \in$* FEMALES *and let $M_t \subseteq$* MALES *the subset of all males that have pulses per call higher than the threshold θ. Then for any female f the worst fitness of an accidental mate for* best-of-n *is equal to the worst fitness of a male $m \in M_f$, while for the minthresh strategy the worst fitness of an accidental mating can be a male $m \notin M_t$ (i.e., any male m with $F(m) < \theta$).*

Proof Assuming a male m_{ac} accidentally mates with f. For *best-of-n*, $m_{ac} \in M_f$ because any accidentally mated male must be closer than the chosen male in M_f, which, in the worst case, has the lowest fitness in M_f. For *minthresh*, the chosen male is the closest with $F(m) \geq \theta$. Hence, any closer male must have worse fitness, hence $m_{ac} \notin Mt$.

The above facts provide a rough qualitative characterization of the differences between the two strategies with respect to accidental matings. We know that accidents can only lower the average mated male fitness, but it is unclear how the two strategies compare quantitatively. Hence, we next describe the experimental setup of our agent-based simulation model that was used to explore the tradeoffs between the two strategies quantitatively.

Experiments

We built our present investigations on our previous agent-based models (Scheutz et al., 2010, 2013), adding various mechanisms for detecting and recording the different types of accidental matings. To briefly summarize the model, we assume that female agents are initially placed at the edges of a rectangular simulated 2D swamp. For simplicity, we assume that male frogs call all the time and never change their call rate or the quality of their call. This male's call is determined by a pulse number and this is the only measure of the fitness of mates (i.e. more pulses are better). We also assume that females can hear and discern the call qualities of all male frogs and make moment-by-moment decision about where to go. Since males never change their calls, a female will move towards a chosen male as long as the male is calling. When a chosen male stops calling because he mated (which is the only reason why males will stop calling in our model), she will pick another male and start moving towards the new male. Whenever any male is with mating distance, both male and female will mate.

Fixed Parameters

We assume a realistic swamp size of $10mx25m$ and a frog size of about $5cm$ in length. Furthermore, we assume each female frog moves at a fixed speed of $1.86cm/sec$ when approaching a male and at $1.44cm/sec$ otherwise. We set the mating range to $4cm$ and always use 25 stationary males placed according to a Gaussian distribution with means in the center of the swamp and standard deviations half the distances to the edges. All females are placed uniformly on the edges of the swamp and always followed a single given strategy with fixed strategy parameters. For details about the simulation model and the simulation update algorithm, which is a straight-forward cycle-based discrete event simulation, see (Scheutz et al., 2010).

Varied Parameters

We vary the number of females – 5, 10, 15, or 20 – initially placed on the swamp's edges. We also vary male call rate based on Gaussian distribution means 6, 12, 18, and 24 with a fixed standard deviation of 2. We consider three different female strategies: *best-of-n* with its parameter $n \in \{1, 2, 3, 4, 5\}$ and *minthresh* with $\theta \in \{6, 12, 18, 24\}$. Finally, we also add a third strategy for comparison, the *random* strategy where a female randomly chooses a male and keeps approaching that male until she either mates with him or needs to pick another random male to approach. Note that the *random* strategy can be used as a baseline compared to the two other strategies because we would not expect any difference in the average fitness of accidentally or non-accidentally mated males when females following the random strategy.

The varied parameters thus span a "parameter space" which we fully explored running 100 simulations with distinct initial conditions for each point in the space for a total of 16000 simulations. The dependent variables were the number of accidental and non-accidental matings as well as the fitness of the accidentally and non-accidentally mated males.

Results

Table 1 shows the overall simulation results for each of the three strategies as well as each strategy parameter for *best-of-n* and *minthresh* averaged of the male call rates and the number of females: column 1 shows the average fitness of mated males, column 2 shows the average fitness of non-accidentally mated males, column 3 shows the average fitness of accidentally mated males, and column 4 shows the frequency of accidental matings.

As can be seen, the random strategy had the highest mean of accidental matings per simulation. Furthermore, although *minthresh* had the lowest mean of accidental matings among the three strategies, it also had the highest influence of accidental matings on the fitness, i.e., when a accident happens, it reduces the fitness drastically.

Strategy	MM	NAMM	AMM	Freq
random	14.901	14.911	14.803	0.940
best-of-n	15.865	15.909	14.474	0.329
best-of-1	14.847	14.847	NaN	0.0
best-of-2	15.813	15.856	14.032	0.297
best-of-3	16.107	16.150	14.725	0.384
best-of-4	16.237	16.300	14.487	0.456
best-of-5	16.319	16.392	14.536	0.509
minthresh	16.985	18.424	10.661	0.210
minthresh 6	15.187	15.206	3.883	0.064
minthresh 12	17.045	18.270	7.272	0.166
minthresh 18	18.397	21.523	10.449	0.259
minthresh 24	18.617	25.267	13.531	0.352

Table 1: Mean Mating (MM), Non-Accidental Mean Mating (NAMM) and Accidental Mean Mating (AMM) fitness and accident frequency for each strategy and parameter.

To compare the main effects of each independent variable on accidental matings, we performed two ANOVAs with *strategy* (s), *number of females* (nf) and *parameter value* (p) as independent variables, and the *number of accidental matings* (nam) and *average fitness* (af) as the dependent variables for each ANOVA. The results of the ANOVA for the dependent variable nam in Table 2 shows a significant main effects of all independent variables on the number of accidental matings, as well as significant two-way and three-way interactions. This confirms the intuitive expectation that using female strategies drastically reduces the average number of accidental matings, the reasons for which we will discuss in the next section.

Of the three two-way interactions, the effect of parameter changes of strategy performs is to be expected – increasing n will increase the average mated male fitness and so that increasing θ, albeit to different degrees (e.g., the increase is generally steeper with *minthresh* because a minimum threshold is imposed, see also (Scheutz et al., 2010)). Similarly, the interactions between strategy parameters and females, strategy and females, and the three-way interactions involving all independent variables are to be expected: an increase in the number of females leads to different increases in average mated male fitness for the three strategies based on the different parameters. However, as the number of females increases, so does the probability of accidental matings. Critically, as shown in Figure 1, the slopes of the two strategies are lower than that of the random strategy, thus confirming that both strategies, *minthresh* and *best-of-n* significantly reduce the number of accidental matings, with *minthresh* overall doing better than *best-of-n*.

The ANOVA in Table 2 shows that there was a significant main effect of the strategy parameter on the number of accidental matings. As shown in Table 1, as the parameter value increases, it also increases the number of accidental matings. Specifically, the value of θ in the *minthresh* strategy determines whether a female can find a potential mate in the swamp or whether she will leave the swamp without mating. Since males are distributed based on a Gaussian distribution centered in the middle of the swamp, it is less likely for females to accidentally mate on their way out of the swamp compared to the chances of accidentally mating based on the *best-of-n* strategy. While we argued previously that no accidental matings are possible with $n = 1$, the average length of the path traversed by each female before she can mate also significantly increases as n increases, and so does the probability of her accidentally mating.

Variable	Df	F value	Pr(>F)
s	1	193.392	<.001
nf	1	1026.013	<.001
p	1	528.230	<.001
s:nf	1	26.981	<.001
s:p	1	602.676	<.001
nf:p	1	113.222	<.001
s:nf:p	1	150.289	<.001

Table 2: ANOVA table for the model "$nam = s * nf * p$" where "nam" is the number of accidental matings, "s" is the strategy, "nf" is the number of females and "p" is the parameter of the strategy, using the best-of-n strategy.

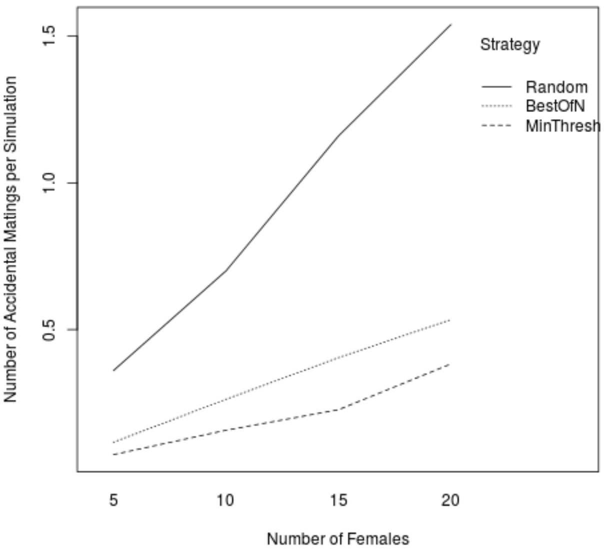


Figure 1: Interaction between the number of females and the number of accidental matings on the three strategies.

Examining the ANOVA for the average fitness of accidental matings (af) in Table 3, there was a significant main effect of the strategy on the results. This suggest that even though strategies reduce the number of accidental matings, the average fitness of the accidentally mated males is lower than that of the non-accidentally mated males as shown in Figure 2. This is as expected because given the strategies' boost to the average mated fitness compared to the random strategy, one would expect accidental matings to yield lower average fitness values in exchange.

In addition, the ANOVA shows a significant main effect of strategy parameter on the average fitness of accidental matings. This can be explained by the difference in fitness of the accidental matings for the *minthresh* strategy, because its parameter θ is an upper limit of the accidental matings (fitness of all accidental matings are less then θ). Thus, increasing the value of the parameter, will also increase the fitness of accidental matings, as shown in Figure 3.

Table 4 shows the influence of the number of females and the parameters of each strategy on the average accidentally mated male fitness. For the *best-of-n strategy* (except for $n = 1$), as the number of females increases, both the fitness of mated males and that of the non-accidentally mated males decrease. This sensitivity to the male-female ratio was originally shown in Scheutz et al. (2010) and is confirmed here: the same patterns emerges even when accidental matings are removed. In contrast, the *minthresh* strategy shows an increase in the fitness of the non-accidentally mated males as the number of females increases (as the changes of finding a higher valued female above the threshold increases with higher numbers of females).

Variable	Df	F value	Pr(>F)
s	1	270.662	<.001
nf	1	0.0082	.928
p	1	246.038	<.001
s:nf	1	4.870	.027
s:p	1	11.285	<.001
nf:p	1	0.164	.685
s:nf:p	1	0.010	.919

Table 3: ANOVA table for the model "$af = s * nf * p$" where "af" is the average fitness of the accidental matings, "s" is the strategy, "nf" is the number of females and "p" is the parameter of the strategy, using the best-of-n strategy.

Using PPC to denote the mean pulses per call and sd the standard deviation of those pulses for each male, occurring an accidental mating when a female is leaving the swamp is dependent of θ, PPC and sd. As shown in Scheutz et al. (2010), the distribution of pulses per call through the males in the swamp is done by a Gaussian distribution. Since the number of pulses per call was defined as an integer, the probability of existing a male with a value of pulses per call lesser than θ, if $\theta = PPC$ is ≈ 40.1. Therefore, if the male-female ratio is greater than 60.9, it is possible to having an accidental mating when the female is leaving the swamp. If $\theta < PPC$, then - for the interval that we tested $(2 * sd)$ - all the females will always mate, therefore no one female will leave the swamp unmated. Finally, if $\theta > PPC$, then - again, for the parameters we tested - no one female will find a mate, therefore all accidental matings happen when the females are leaving the swamp.

Table 5 shows the influence of mean pulses per call on the fitness of the matings without remove the accidental matings and also after removing them. Again, changing the mean pulses per call does not have any influence in the *best-of-n strategy*, only increasing its value in 6. On the other hand, *minthresh* is influenced by mean pulses per call, because, as shown previously, the probability of finding a mate using this strategy is dependent of the value of its parameter and also the mean pulses per call. However, if you fix the value of θ and vary the value of PPC, maintaining the relation $\theta < PPC$, the only difference will be an increment of 6, as in the *best-of-n strategy*. For example, for $\theta = 6$ and $PPC = 12$, we have the fitness of mated equals to 11.846, if we look over $PPC = 18$, the same fitness of mated is 17.846, an increment of 6.

Discussion

It is a known fact that female treefrogs do not try to avoid males from other species of treefrogs (Gerhardt et al., 1994),

Strategy	5 Females		10 Females		15 Females		20 Females	
	MM	**NAMM**	**MM**	**NAMM**	**MM**	**NAMM**	**MM**	**NAMM**
random	14.851	14.865	14.851	14.861	14.935	14.941	14.968	14.979
best-of-1	14.788	14.788	14.836	14.836	14.861	14.861	14.903	14.903
best-of-2	16.171	16.215	15.889	15.937	15.716	15.762	15.475	15.512
best-of-3	16.607	16.658	16.289	16.328	15.967	16.014	15.565	15.599
best-of-4	16.830	16.900	16.471	16.541	16.075	16.133	15.571	15.625
best-of-5	17.014	17.103	16.574	16.658	16.102	16.163	15.586	15.644
minthresh 6	15.150	15.165	15.180	15.192	15.202	15.216	15.217	15.249
minthresh 12	17.724	18.236	17.130	18.255	16.850	18.280	16.542	18.308
minthresh 18	20.034	21.501	18.596	21.505	17.978	21.531	17.361	21.553
minthresh 24	21.493	25.281	18.909	25.242	18.033	25.270	17.258	25.273

Table 4: Mean Mating (MM) and Non-Accidental Mean Mating (NAMM) fitness for each strategy parameter and number of females in the swamp.

Strategy	6 PPC		12 PPC		18 PPC		24 PPC	
	MM	**NAMM**	**MM**	**NAMM**	**MM**	**NAMM**	**MM**	**NAMM**
random	5.918	5.929	11.896	11.906	17.896	17.906	23.896	23.906
best-of-1	5.851	5.851	11.846	11.846	17.846	17.846	23.846	23.846
best-of-2	6.816	6.859	12.812	12.855	18.812	18.855	24.812	24.855
best-of-3	7.111	7.154	13.106	13.148	19.106	19.148	25.106	25.148
best-of-4	7.241	7.305	13.235	13.298	19.235	19.298	25.235	25.298
best-of-5	7.325	7.398	13.317	13.390	19.317	19.390	25.317	25.390
minthresh 6	7.211	7.284	11.846	11.846	17.846	17.846	23.846	23.846
minthresh 12	5.963	12.667	13.210	13.285	17.846	17.846	23.846	23.846
minthresh 18	5.664	NaN	11.963	18.667	19.210	19.285	23.846	23.846
minthresh 24	5.664	NaN	11.664	NaN	17.963	24.667	25.210	25.285

Table 5: Mean Mating (MM) and Non-Accidental Mean Mating (NAMM) fitness for each strategy parameter and pulses per call (PPC).

thus sometimes inter-species accidental matings occur; however, often the hybrids are sterile (Johnson, 1963) or do not survive until sexual maturity (Schlefer et al., 1986). Although we modeled the simulation with only one species of treefrogs, the negative influence of accidental matings is also sustained on observations of their behavior in nature. Using strategies to select a mate is, therefore, overall beneficial in that it can drastically reduce the probability of an accidental mating to occur, as shown by our results. Moreover, while the *best-of-n* strategy leads to more accidental matings than the *minthresh* strategy, the accidents that happen using *minthresh* have a higher influence on the average mated male fitness than those that occur with *best-of-n*. Hence, we next analyze how and why these accidents occur.

Analysis: The Nature of Accidents

Figure 1 showed that the number of females has an influence on the number of accidental matings and that the use of a strategy to choose a male to mate significantly reduces the number of accidental matings, which we call the *strategy effect*. Furthermore, we call the fact the *minthresh* strategy has a lower number of accidental matings compared to the *best-of-n strategy* the *minthresh effect*.

Both effects can be explained by considering the probability P_a of an accidental mating to occur. In order to calculate P_a, we can define a vector L_f which in its i-th position contains 1 if and only if another male is on the path of f to the male m_i and 0 otherwise. To generate this vector, we can trace straight lines from f to all males in the environment and verify if the line intersects another male (within mating range). Let P^i_{random} be the probability of a male i be chosen by the random strategy, $P^i_{random} = \frac{1}{|MALE|}$ for all males in the swamp. Let $P^i_{bestofn}$ be the probability of a male in position i be chosen based on the parameter n on the *best-of-n* strategy. Finally, let $P^i_{minthresh}$ be the probability of a male i be chosen based on the parameter θ from the *minthresh* strategy. Generically, we denote $P^i_{strategy}$ the probability of

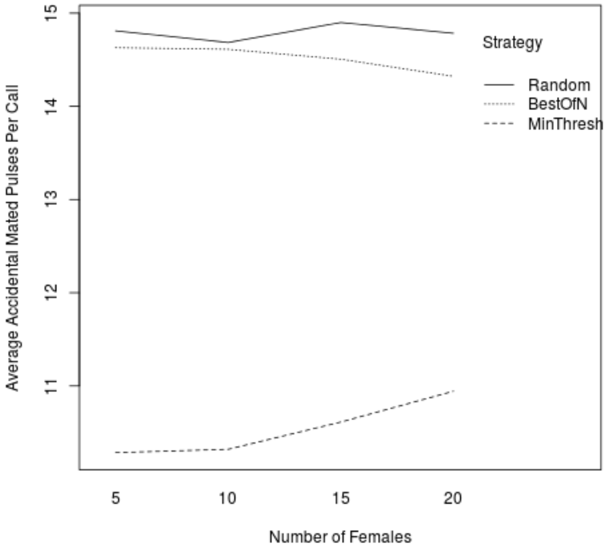


Figure 2: Interaction between the number of females and the fitness of the accidental matings on the three strategies.

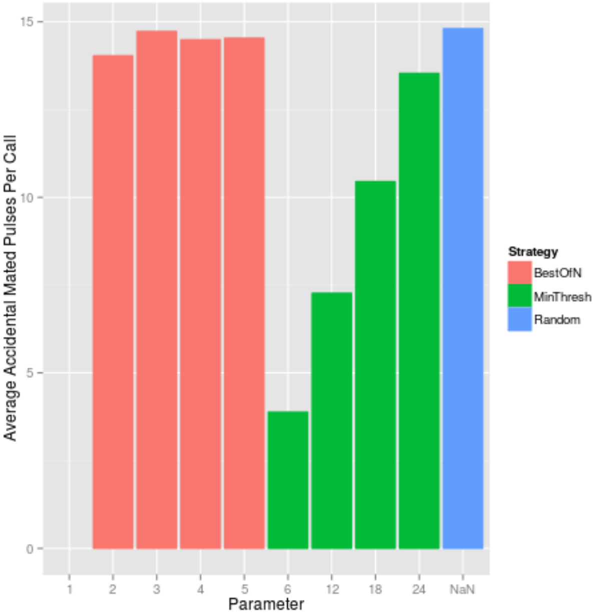


Figure 3: Interaction between the parameters and the fitness of the accidental matings on the three strategies.

male i to be chosen by a *strategy*.

Given the state of the swamp at any point in time, the probability of a female accidentally mating with a male during the simulation is determined by $P_a = \sum_{i=1}^{|MALE|} (P^i_{strategy} \cdot L_f[i])$.

To elucidate the difference in P_a for distinct strategies, we consider two configurations of the swamp with just a slight change on the position of one male, as shown in Figure 4(a) and in Figure 4(b). Both figures contain one female f at the bottom and the three males closest to f. Let the fitness of the three males be $F(m_1) > F(m_2) > F(m_3)$ and let $D(f, m_{leftmost}) > D(f, m_{rightmost} > D(f, m_{central})$. The number of distinct arrangements of the three males with the different different fitness values is $3! = 6$.

Now consider the configuration in Figure 4(a). First, we need to define the vector L_f which represents the existence of another male on the path toward the chosen one. Let the first position contain the leftmost male frog, the second position contains the rightmost male frog and the third filled with the central frog in the environment. Thus, $L_f = \{0, 1, 0\}$.

For the *random* strategy, it is clear that, independently of which male has the best fitness, $P^i_{random} = \frac{1}{3}$. Consequently, $P_a = (\frac{1}{3} \cdot 0) + (\frac{1}{3} \cdot 1) + (\frac{1}{3} \cdot 0) = \frac{1}{3}$.

For the *best-of-n* strategy, we need to define the parameter n in order to calculate the probability of an accidental mating to occur. For $n = 1$, we showed in Fact 1 that accidental matings are impossible. Hence, for $n = 2$, we can calculate the value of $P^i_{bestofn}$. First, selecting the leftmost frog, the probability of him being chosen is 0 because there are two other males that are closer to f. Now, selecting the rightmost frog, there are three distinct arrangements out of six possible ones in which he will be chosen: $(\{m_1, m_2, m_3\}, \{m_2, m_1, m_3\}, \{m_3, m_1, m_2\})$. Therefore, the probability of the rightmost being chosen is $\frac{1}{2}$. Finally, there are three distinct arrangements in which the central frog will be chosen $(\{m_1, m_3, m_2\}, \{m_2, m_3, m_1\}, \{m_3, m_2, m_1\})$, consequently the probability of the central frog being chosen is also $\frac{1}{2}$. As a result, for $n = 2$, $P_a = (0 \cdot 0) + (\frac{1}{2} \cdot 1) + (\frac{1}{2} \cdot 0) = \frac{1}{2}$. For $n = 3$, the probability of the rightmost being chosen is $\frac{1}{3}$, as is the one for the leftmost and the central one. Therefore, for $n = 3$, $P_a = (\frac{1}{3} \cdot 0) + (\frac{1}{3} \cdot 1) + (\frac{1}{3} \cdot 0) = \frac{1}{3}$.

On the other hand, for the *minthresh strategy*, we define $\theta = F(m_2)$, thus only two males have a fitness greater or equal than the threshold. Selecting the leftmost frog, the probability of him being chosen is 0 because for every male there exists at least one more with a fitness value greater or equal to the threshold, but that is closer to the female than the leftmost. Choosing the rightmost, we have two arrangements in which he will be picked: $(\{m_1, m_2, m_3\}, \{m_2, m_1, m_3\})$, thus the probability of him being chosen is $\frac{1}{3}$. Lastly, the central frog has a probability of $\frac{2}{3}$ of being chosen, or four arrangements: $\langle m_1, m_3, m_2\rangle, \langle m_2, m_3, m_1\rangle, \langle m_3, m_1, m_2\rangle, \langle m_3, m_2, m_1\rangle$. Therefore, $P_a = (0 \cdot 0) + (\frac{1}{3} \cdot 1) + (\frac{2}{3} \cdot 0) = \frac{1}{3}$.

In sum, $P_{a,random} = \frac{1}{3}$, $P_a^{n=2} = \frac{1}{2}$, $P_a^{n=3} = \frac{1}{3}$, $P_{a,minthresh} = \frac{1}{3}$. However, if we look at Figure 4(b), although the $P_{strategy}$ values are the same for every strategy, we need to define a new vector $L_f = \{1, 0, 0\}$, as a consequence, the probabilities P_a can be different. Hence, we get $P_{a,random} = (\frac{1}{3} \cdot 1) + (\frac{1}{3} \cdot 0) + (\frac{1}{3} \cdot 0) = \frac{1}{3}$, $P_a^{n=2} = (0 \cdot 1) + (\frac{1}{2} \cdot 0) + (\frac{1}{2} \cdot 0) = 0$, $P_a^{n=3} = (\frac{1}{3} \cdot 1) + (\frac{1}{3} \cdot 0) + (\frac{1}{3} \cdot 0) = \frac{1}{3}$ and $P_{a,minthresh} = (0 \cdot 1) + (\frac{1}{3} \cdot 0) + (\frac{2}{3} \cdot 0) = 0$.

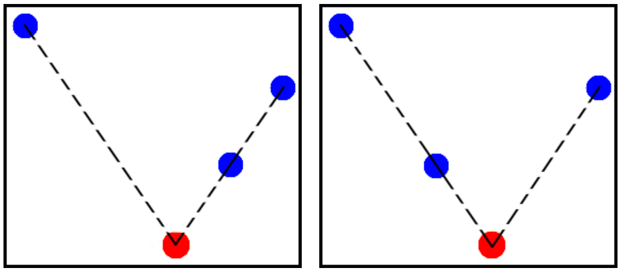


Figure 4: Two examples of different swamp states. The dashed lines represent distinct trajectories of the female treefrog and are used to create the L_f vector.

We now can compare the mean of the probabilities of accidental matings in the two configurations. For the random strategy, the mean will be $\frac{1}{3}$, for the *best-of-n* strategy using $n = 2$ the mean will be $\frac{1}{4}$ and using $n = 3$ the mean will be $\frac{1}{3}$. And for the *minthresh* strategy, the mean will be $\frac{1}{6}$. Therefore $P_{a,random} = P_a^{n=3} > P_a^{n=2} > P_{a,minthresh}$. Although the results of random and $n = 3$ were the same, it is clear that if the number of males in the swamp was greater than n, $P_{a,random}$ will be greater than any n showing then the *strategy effect*. If we compare the probabilities P_a^n and $P_{a,minthresh}$ we can verify the *minthresh effect* as well.

Conclusion and Future Work

Accidental matings happen in real environments where females end up with males they did not choose. In this paper, we have investigated the frequency and changes in mated male fitness in accidental matings specifically in the context of the female choice of the gray treefrogs *hyla versicolor* based on two main strategies hypothesized to be widely used in nature. Our simulation results showed that the *best-of-n* strategy had a less preferable, higher incident rate of accidental matings compared to the *minthresh strategy*, while also having a preferable, higher average mated male fitness. We demonstrated how and why these two dimensions trade off both based on strategy parameters and on the male-female ratio in the swamp when males are distributed in the swamp according to a Gaussian distribution.

As a next step, we intend to investigate how different male distributions in the swamp could have an impact on the results, i.e., in particular on the probability of females accidentally bumping into a non-chosen male. For example, we would expect that different distributions might change the relative likelihood of accidental matings among the strategies (e.g., a uniform or inverse Gaussian male distribution might be an equalizer between *best-of-n* and *minthresh strategy* regarding the frequency of accidents). We are also interested in evaluating the accident rates when males are allowed to reposition themselves as previously investigated in Scheutz et al. (2013). For example, it is currently unclear where repositioning will increase or decrease accidents, but the overall expectation is that accidents can be increased based on the positioning strategy chosen by the males. In particular, one would expect that so-called "satellite males", i.e., males that do not call at all, might use a strategy that favors accidental matings (which is in their favor because they cannot be detected otherwise). Finally, it would be interesting to derive more detailed general principles about strategy-dependent accidental matings that might inform theories of female choice independent species-specific details.

Acknowledgements

This work was supported in part by NSF grant #1257815.

References

Baugh, A. T. and Ryan, M. J. (2009). Female tungara frogs vary in commitment to mate choice. *Behavioral Ecology*, page arp120.

Gerhardt, H. C. (1994). Selective responsiveness to long-range acoustic signals in insects and anurans. *American zoologist*, 34(6):706–714.

Gerhardt, H. C., Dyson, M. L., Tanner, S. D., and Murphy, C. G. (1994). Female treefrogs do not avoid heterospecific calls as they approach conspecific calls: implications for mechanisms of mate choice. *Animal Behaviour*, 47(6):1323–1332.

Janetos, A. C. (1980). Strategies of female mate choice: a theoretical analysis. *Behavioral Ecology and Sociobiology*, 7(2):107–112.

Jennions, M. D. and Petrie, M. (1997). Variation in mate choice and mating preferences: a review of causes and consequences. *Biological Reviews*, 72(2):283–327.

Johnson, C. (1963). Additional evidence of sterility between call-types in the hyla versicolor complex. *Copeia*, pages 139–143.

Scheutz, M., Harris, J., and Boyd, S. K. (2010). How to pick the right one: Investigating tradeoffs among female mate choice strategies in treefrogs. In *Proceedings of the Eleventh International Conference on Simulation of Adaptive Behavior - SAB 2010*, Paris - Clos Luc, France.

Scheutz, M., Smiley, M., and Boyd, S. (2013). Exploring male spatial placement strategies in a biologically plausible mating task. In *IEEE Symposium on Artificial Life*.

Schlefer, E. K., Romano, M. A., Guttman, S. I., and Ruth, S. B. (1986). Effects of twenty years of hybridization in a disturbed habitat on hyla cinerea and hyla gratiosa. *Journal of herpetology*, pages 210–221.

Welch, A. M., Semlitsch, R. D., and Gerhardt, H. C. (1998). Call duration as an indicator of genetic quality in male gray tree frogs. *Science*, 280(5371):1928–1930.

Incremental Neuroevolution of Reactive and Deliberative 3D Agents

Adam Stanton[1] and Alastair Channon[1]

[1]School of Computing and Mathematics, Keele University, ST5 5BG, UK
{a.stanton,a.d.channon}@keele.ac.uk

Abstract

Following earlier work on the neuroevolution of deliberative behaviour to solve increasingly challenging tasks in a two-dimensional dynamic world, this paper presents the results of extending the original system to a three-dimensional rigid body simulation. The 3D physically based setting requires that a successful agent continually and deliberately adjust its gait, turning and other motor control over the many stages and sub-stages of these tasks, within its individual evaluation. Achieving such complex interplay between motor control and deliberative control, within a neuroevolutionary framework, is the focus of this work. To this end, a novel neural architecture is presented and an incremental evolutionary approach used to bootstrap the locomotive behaviour of the agents. Agent morphology is fixed as a quadruped with three degrees of freedom per limb. Agent populations have no initial knowledge of the problem domain, and evolve to move around and then solve progressively more difficult challenges in the environment using a tournament-based co-evolutionary algorithm. The results demonstrate not only success at the tasks but also a variety of intricate lifelike behaviours being used, separately and in combination, to achieve this success. Given the problem-agnostic controller architecture, these results indicate a potential for discovering yet more advanced behaviours in yet more complex environments.

Introduction

Living systems exhibit a large variety of coordinated activities at many different scales. We find homeostasis, locomotion, learning, group and social behaviours throughout the natural world. Since the earliest days of Artificial Life, a defining ambition has been to understand how to engineer systems that exhibit some of these complex behaviours, either to solve problems or to understand the underlying principles that gave rise to them in nature (Langton, 1989).

The specification of a model requires assumptions to be made concerning the degree to which its most basic units and the rules governing their behaviour are able to act as reliable proxies for their natural analogues. The granularity of a system has a direct impact on both its speed and its potential to accurately mimic nature, and on the strength of conclusions about the natural world based on phenomena observed to emerge from interactions within it.

The Dimensionality of Virtual Environments

Simulations of living systems have covered a broad range of abstraction but typically aim to exhibit behaviours at the level above that specified in the model's design. When building animat simulations that focus on interactions recognisable at the human scale, such as moving around, fighting and environmental manipulation, one of the key distinctions between designs is the physicality in which agents operate, specifically the choice between 2D and 3D environments. A two-dimensional world abstracts simulations away from the natural physical domain. Agents in these flat environments generally do not have to solve any complex physical control problems (Channon and Damper, 1998), as controllers are able simply to signal directions in which to move or turn the agent. Such models can encourage early emergence of more complex composite behaviours but preclude the development of novel motor control which may later allow for a richer interaction between agents and their environments. Two-dimensional simulations have not tended toward clearly *displaying* the prodigal physical interaction observed in nature, whether or not complex (simulated) non-physical interactions have evolved. This can be attributed, at least in part, to the fundamental rigidity and paucity of physical actions in such environments.

By contrast, having three-dimensional articulated bodies in a 3D world provides for much greater intricacy in how agents can interact with their environment and each other. Agents must begin to construct a coordinated motor pattern that results in basic directional motion before richer behaviours can develop as composites of these lower-level patterns. The specific characteristics of the environment are implicitly included in the performance of these motor patterns, and this couples agents to their environment. This coupling is crucial, together with the coupling of brain and body, to two key principals of embodied cognition: "first that cognition depends upon the kinds of experience that come from having a body with various sensorimotor capacities, and second, that these individual sensorimotor capacities are themselves embedded in a more encompassing biological, psychological and cultural context" (Rosch et al., 1991). Recent trends reinforce this point of view, highlighting the importance of morphology and soft materials in the embodied loop (Pfeifer et al., 2014).

In terms of the ongoing ambition to evolve advanced life-like behaviour, both 2D and 3D approaches have been fruitful. For example, using 2D non-articulated agent bodies, early work by Yaeger showed (in a 3D environment) the emergence of complex collective behaviour (Yaeger, 1993); Channon demonstrated the first candidate synthetic open-ended evolutionary system using an agent-based (2D) world (Channon and Damper, 1998); and Robinson et al. (2007) evolved agents capable of reactive and deliberative behaviours in novel and dynamic environments.

In 3D the inherent complexities of articulated 3D physical form refocused work on the problems of motor control and locomotion: difficulties that had been largely abstracted away in 2D models. The seminal work by Sims (1994) remains an exemplar to the present day. Subsequent research has made incremental steps from this point, including demonstrating realistic co-adapted behaviours using just general purpose neurons (Miconi and Channon, 2006), making use of a human-specified syllabus of reactive locomotion-based tasks (Lessin et al., 2013) and using Novely Search (Lehman and Stanley, 2008) to evolve a range of gaits for a fixed morphology robot (Cully and Mouret, 2015), but continues to focus primarily on locomotion alone, leaving more complex behaviours aside.

General Approach of this Work

This work constitutes a first attempt to combine the incremental neuroevolution of reactive and deliberative behaviours with the neuroevolution of a 3D agent's motor control. Our overarching aim is the incremental evolution of sophisticated behaviours, for the population to overcome increasingly complex challenges in the agents' environment over evolutionary time.

The challenge is difficult because deliberative behaviour will be limited by necessary performance in motor control. An incremental approach can take this subtask-interdependency into account and prevent loss or lack of evolutionary gradient early in evolution. However, Stanton and Channon (2013) found that care is required when designing such incremental steps, as changing selection pressures too rapidly or too slowly can, respectively, cause evolution to lose gradient or over-fit to the current challenge. That work also demonstrated that it is necessary to revisit earlier incremental steps in order to prevent the loss of evolved abilities and therefore to find general solutions.

There is then a question of how to implement deliberative processing alongside physical control in a single controller. Deliberative planning systems learn a state-based action policy in order to select the best next state given a set of available actions. In contrast, flexible control of 3D motion requires a continuous-time closed-loop control system to keep physical variables within operational parameters. Also, for locomotive behaviours, a self-generating oscillation within the controller or body–controller action loop is necessary to achieve a reliable gait.

The requirements of each of these control systems is fundamentally different; it is difficult to design an architecture that can effectively learn the two different problems. The choice is between either an architecture that is general enough to be capable of both episodic categorisation and time-based close-coupled motor control, or a combination of the two architectures each tailored to a specific part of the problem and integrated elsewhere. In this work we opt for the latter, as a pragmatic step toward a more general architecture.

Hypothesis

The present work examines the following hypothesis: *that it is possible to produce reactive, deliberative behaviours in three-dimensional virtual creatures using a general evolutionary paradigm to optimise an implementation of the hybrid neural architecture detailed below*. The "River Crossing" (RC) task devised by Robinson et al. (2007) is used as the baseline reactive–deliberative problem. This task is adapted by the addition of a requirement of physical motor control in 3D, and the complete problem against which agents are tested is hereafter referred to as the 3D River Crossing or 3D RC task.

The remainder of this paper presents details of the 3D RC task, the agent and its hybrid neural architecture, and the evolutionary system, before reporting qualitative and quantitative results and our conclusions. It provides an existence proof that demonstrates the sufficiency and overall success of the design.

Experimental Design

The main contribution of this paper is the novel fusion of multiple neural architectures, each addressing different aspects of the 3D RC task, in order to enable the incremental evolution of agents that achieve the full task. This section of the paper introduces the environment and physical model and then describes the hybrid neurocontroller in detail, making reference to the inputs and outputs defined by the agent–environment relationship. Finally, the evolutionary algorithm is described in terms of the parameters of the neural architecture, and the experimental set-up is outlined.

Environment and Physical Model

The environment for the evolutionary problem is a modified version of the RC task first used in Robinson et al. (2007). In this task, agents exist and move around in a discrete, 20×20 bounded grid world. Each grid cell has attributes which can affect the agent: *traps* kill it, as does *water* (drowning); *grass* is neutral and *stones* can be picked up and put down. Stones can be placed on water, enabling bridges to be built. The final attribute, *resource*, is the agent's goal. The RC task is an incrementally difficult challenge, with a staged introduction of difficulties. By collecting the resource, agents progress through more complicated environments, eventually arriving at a 20×n-cell river, where n is the increasing width of the river and thus the difficulty of the bridge-building task.

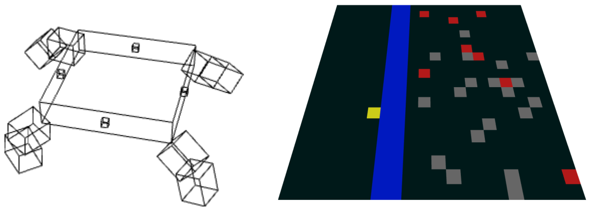


Figure 1: Agent morphology and environment, showing resource in yellow, river in blue, traps in red and stones in grey.

The 3D RC environment used in this work extends the 2D RC environment. Agents have a symmetrical quadruped body plan (figure 1) comprised of a torso (dimensions 1.0×1.0×0.2 cell-widths), four upper limbs (0.5×0.2×0.2), four lower limbs (0.5×0.2×0.2) and four small sensors (0.05×0.05×0.05). The upper limbs are attached to the torso at each lower corner with a 2-axis constraint. The constraint limits the range of motion of the upper limb relative to the torso, to $\frac{\pi}{2}$ radians around the vertical axis, and π radians around the line lying tangent to the agent's torso in the plane of the torso. Lower limbs are connected to upper limbs via a knee constraint which limits the range of motion between the two parts to $\frac{\pi}{2}$ radians around the y-axis. The sensors are attached with fixed constraints to the centre of each of the four faces of the agent's torso perpendicular to the ground plane. The physical simulator used was Open Dynamics Engine (ODE) version 0.13.1, with friction pyramid approximation for contact response ($\mu = 10.0$) between agent and the ground plane, universal ERP of 0.2 and CFM of 5×10^{-5}.

In order to bootstrap the evolution of locomotive behaviour, two additional levels were added at the start of the incremental RC task. The first level distributes "food" around the RC world. This confers additional fitness on agents once collected. The second level ("dash") has only one occupied cell, containing the resource. These levels together promote locomotive behaviour, and ultimately optimise the behaviour for speed of movement.

The difficulty of the RC environment is increased incrementally across six progressively more challenging levels. An agent's fitness is incremented from zero by 100 each time it successfully finds the resource, a requirement to progresses to the next level.

- Level 1: *Food*. The RC environment contains only cells with the resource (one cell) and food (probability 1/20 per cell). Interaction with a food cell removes the food from the environment and increments the agent's fitness by 1.
- Level 2: *Dash*. This level contains only a single resource cell which agents must discover.
- Level 3: *Stones and Traps*. This level contains eight traps and twenty stones, as well as the target resource.
- Level 4: *Easy bridge*. This level is as level three but with a river of width 1 crossing the terrain.
- Level 5: *Medium bridge*. As level four, but width 2.
- Level 6: *Hard bridge*. As level five, but width 4.

On completion of level 6, agents are returned to level 1 and can continue to accumulate fitness until the time limit of 10 simulated minutes is reached, when evaluation is terminated.

Neural Architecture

A neural architecture capable of solving the 2D RC task was a major contribution of Robinson et al. (2007) and is extended in the present work. In the 3D RC task, an agent's neurocontroller transforms sensory inputs into torque values for motor control, which gives rise to behaviour in the physically simulated environment. The control system must produce directed locomotive behaviour in the quadruped, and change locomotive behaviour over the stages and sub-stages of the RC task, according to external (sensory) and internal (neural) state.

The hybrid neural architecture (figure 2) integrates the outputs of the RC world *decision network* (DN) and the diffusive *shunting model* (SM) with the inputs of the *physical network* (PN), and then use this information to pilot the agent through the world by affecting the operation of the agents' *pattern generator* (PG) neurons.

The Decision Network. The DN architecture follows the design laid out in Robinson et al. (2007). The DN is a standard feedforward neural network which takes inputs representing the attributes of the agent's current location in the RC world, and an input indicating whether or not the agent is currently carrying a stone. The hidden layer contains four neurons which sum over the inputs and apply a hyperbolic tangent activation function. The output layer sums over the hidden layer, applies a hyperbolic tangent activation function and tests at the thresholds -0.3 and 0.3; output neurons have three possible values: -1, 0 or 1, and determine the *iota values* used in the SM. These iota values indicate the saliency of the attributes in the environment, so the DN outputs iota values for each attribute (resource, stone, water and trap) except grass (which has an iota value of zero).

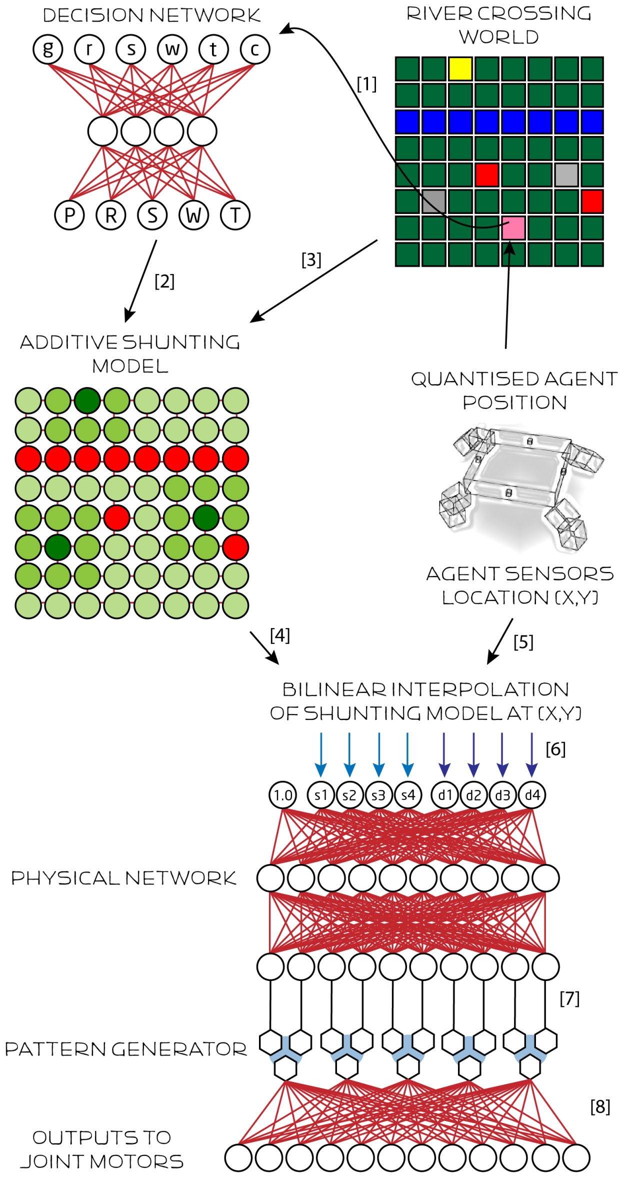


Figure 2: Neural architecture. Attributes at the agent's position (g=grass, r=resource, s=stone, w=water, t=trap, c=carrying flag) determine inputs to the Decision Network [1]. The Shunting Model constructs a landscape using iota values output by the DN [2] (P=pickup action, R=resource, S=stone, W=water, T=trap) and the locations of objects [3]. The SM activity landscape is interpolated [4] at the positions of the animat's four sensors [5], and these values fed to the Physical Network [6]. PN outputs are fed to the Pattern Generator Network [7], which outputs to neuromotor controllers. Links in red are genetically specified.

The Shunting Model. The SM was first used as a novel approach to motion planning by Meng and Yang (1998). The approach uses the homomorphism between the varying external environment and the intrinsic dynamics of the architecture to achieve route generation (planning) without explicitly searching over possible paths. It is a generalisation of the potential field approach of Glasius et al. (1995), historically an evolution of the model of neural connectivity first proposed in Hodgkin and Huxley (1952). The SM uses a locally-connected, topologically-organised network of neurons to propagate desirable states across the entire network of transitions in the space. This produces an *activity landscape* with peaks at target states and valleys at configurations to avoid. One of the most common implementations of the SM is the *additive model* (Grossberg, 1988), which sacrifices gain control (and thus, stability) for simplicity. This model defines the following differential equation to model the diffusion of input values across the state landscape:

$$\frac{dx_i}{dt} = -Ax_i + \sum_{j \in N_i} w_{ij}[x_j]^+ + I_i \tag{1}$$

where each neuron in the SM corresponds to one discrete cell in the environment; x_i is the activation of neuron i, taken to be zero outside of the environment; A is a passive decay rate; N_i is the receptive field of i; w_{ij} is the connection strength or weight from neuron j to neuron i, specified to be set by a monotonically decreasing function of the Euclidean distance between cells i and j (zero outside of the neighbourhood); the function $[x]^+$ is $max(0, x)$; and I_i is the external input to neuron i.

This technique was used in Robinson et al. (2007) to model the state space of the RC problem by directly representing the discrete RC world in the configuration of the SM, with each cell's receptive field set to be the eight cells in its Moore neighbourhood, within which all $w_{ij} = w$, and external input I_i determined by the attributes present in cell i and the saliency (*iota value*) for those attributes as computed by the DN. Neural activations propagate from external input I according to the local connectivity of the neurons, and the entire network can be considered a diffusive model that produces landscapes in which following positive gradients leads to target states. With well-chosen constant multipliers, this method exhibits no undesirable dynamics and has been found to be considerably versatile in a variety of subsequent works, including those of Borg et al. (2011) and Luo et al. (2014).

In this work, we simplify and clarify the setting of of decay rate and scales for distance (or weights) and iota values. A stable solution ($x_i^{new} = x_i$ for all i) to equation 2 is a stable solution ($\dot{x} = \mathbf{0}$) to equation 1. We absorb the constant A into the scales for iota values and distances, and set and limit weights and activation according to neighbourhood size (8) and maximum iota value ($maxI$=15), resulting in equation 3.

$$x_i^{new} = \frac{1}{A}\left(\sum_{j \in N_i} w_{ij}[x_j]^+ + I_i\right) \quad (2)$$

Following the computation of external inputs I by the DN, we zero SM activations and then iterate equation 3 fifty times to allow activity to propagate and stabilise across the 20×20 array of SM neurons.

$$x_i^{new} = min\left(\frac{1}{8}\sum_{j \in N_i}[x_j]^+ + I_i,\ maxI\right) \quad (3)$$

The Physical Network. The PN controls the agent's behaviour in the world. It receives as inputs the SM activations (interpolated) at the positions of the four sensors located on the four sides of the agent's torso. Since the SM represents a neural quantisation of the continuous landscape in which the sensors move, a single value is calculated for each sensor using a bilinear interpolation of the SM's activity values at the four points around the relevant sensor:

$$\begin{aligned} a(x,y) \quad = \quad & f[\lfloor x \rfloor, \lfloor y \rfloor](1-\{x\})(1-\{y\}) + \quad (4)\\ & f[\lceil x \rceil, \lfloor y \rfloor]\{x\}(1-\{y\}) + \\ & f[\lfloor x \rfloor, \lceil y \rceil](1-\{x\})\{y\} + \\ & f[\lceil x \rceil, \lceil y \rceil]\{x\}\{y\} \end{aligned}$$

where $a(x,y)$ is the interpolated activity at $(x,y) \in \mathbb{R}^2$, $f[i,j]$ is the SM activation at the discrete point $(i,j) \in \mathbb{Z}^2$ and {x} denotes the fractional part of x.

These four sensor values are normalised (divided by $maxI$) and then fed into the PN, together with four values that indicate which sensor has the maximum value. The PN operates as a standard feedforward neural network where hidden nodes receive a weighted sum of the inputs. The hidden layer uses a hyperbolic tangent activation function in order to maintain negative values. The output layer uses a sigmoid activation function.

The Pattern Generator Network. The PG is a set of pre-evolved oscillatory neural circuits which are modelled on the networks of leaky integrators presented in Beer and Gallagher (1992) and used for locomotor pattern generation in many subsequent works, including Reil and Husbands (2002) and Stanton and Channon (2013). The circuits themselves are three-neuron motifs evolved to produce 1Hz sinusoidal oscillations from an output node in the presence of an input signal, and to be quiescent otherwise. Each complete PG network has a set of five identical motifs, initially isolated, which receive input from the PN via a set of weights and send their outputs to the final stage of the agent's controller. The neurons comprising these motifs are simple continuous-time leaky integrators, with behaviour governed by the following equations:

$$\tau_i \frac{dA_i}{dt} = -A_i + \sum_{j=0}^{n} w_{ij} O_j \quad (5)$$

$$O_i = tanh(\frac{\alpha_i - A_i}{2}) \quad (6)$$

where A_i is the activation of a neuron i, O_i is the output of neuron i, w_{ij} is the weight from neuron j to neuron i, α_i is the bias of neuron i and τ_i is the time-constant of neuron i. At each iteration of the update algorithm ($dt = 0.01s$), equation 5 computes the change in the activity of the *i*th neuron for all neurons, and then equation 6 computes the output value for all neurons. It is this output value that is used by the neuromotor controllers.

To generate the original motif, a population of 1000 randomly initialised three-neuron networks was created with weights, time-constants and biases defined by a real-valued genotype. These networks were evaluated against a fitness function which measured the match between the desired frequency and the output response by summation of the undesirable (non-target) frequencies found in the frequency domain after application of Fourier transform. Networks were simulated for 10 seconds, twice. Once with a high input and a target frequency of 1Hz, and once with no input and a target quiescent state. Through three-genome tournament selection, strong candidates were used to generate new, mutated members of the population using the same evolutionary parameters as the general system described below.

Neuromotor Controllers. In the final stage, 12 motor controllers (one for each degree of freedom in the agent's morphology) receive the outputs of the PG network via a weighted sum and sigmoid activation function. These motor controllers implement a proportional-derivative (PD) controller, as used by Reil and Husbands (2002), which takes network outputs to be target angles within each joint's range of motion and applies a torque to the joint according to the following formula:

$$T = k_s(\theta_d - \theta) - k_d\dot{\theta} \quad (7)$$

where T is the torque applied to the joint, k_s is the spring constant, k_d is the damping constant, θ_d is the target angle and θ is the current angle. In this work, $k_s = 0.25$ and $k_d = 0.175$ were found to produce stable action at joints. This method has the advantage of relieving the neurocontroller of the problem of balancing an agent's weight against the force of gravity.

Evolutionary System

A steady-state evolutionary algorithm was used, in which a population of 150 agents are evaluated in groups of three and the least-fit individual replaced by a mutated single-point crossover of the fitter two.

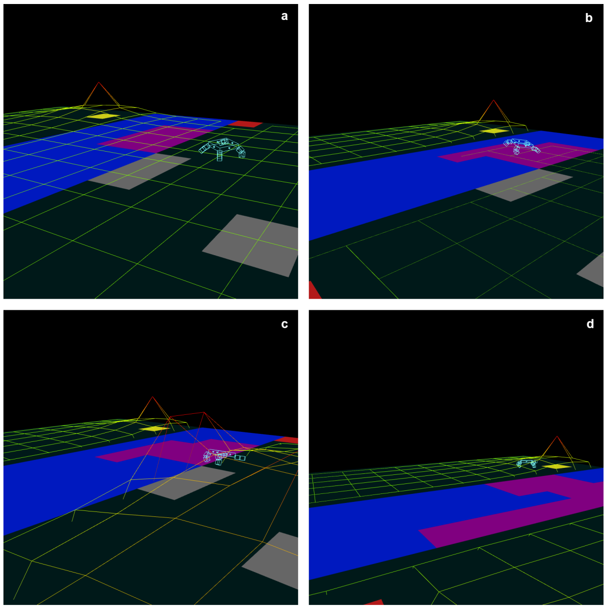


Figure 4: Bridge building in action. In (a) the agent has already started to build a bridge and is returning to collect another stone. In (b) the agent has just dropped a stone and is beginning to turn around. In (c) the agent is carrying a stone to drop on the water. In (d) the agent has completed the bridge and is about to reach the resource. The figure also illustrates the SM activity landscape superimposed on the 3D RC world and shows the changes to this landscape due to the updated iota values that occur as the agent's state, and thus DN inputs, vary.

Genetic Representation. Individuals' neurocontrollers are represented as an array of floating-point values. The sections are laid out as arrays of weights for each network stage as outlined above: the DN input–hidden and hidden–output weights, the PN input–hidden and hidden–output weights, the PG interneuron weights and the PG–motor weights.

Results

Twenty runs were carried out, each for 10^6 tournaments.

Qualitative Results. In those runs scoring highly on the final level of the task, intricate and diverse behaviours can be observed as the agents progress through their environmental challenges. In any single species, several different locomotive strategies can be observed depending on whether the agent is near or far from its target, and whether there are obstacles in the way. In the case of a "clear run", agents often gallop (figure 3) toward the target, whereas if more careful movement is required agents will progress more slowly, making time to avoid unexpected sensory conditions (i.e. traps and water). In both cases, directed control is observed as agents update their heading whilst engaging in locomotion to remain aligned with the target. Agents also often display a distinct "turning" behaviour which will engage if the agent is beyond some angular threshold away from facing its target. Figure 4 shows an example evolved agent solving 3D RC task.

One of the most lifelike behaviours to be observed is avoidance: due to the non-spreading negative values in the activity landscape agents can unexpectedly encounter a highly negative region. In this case, agents will often crouch and spring back from the hazard, minimising the chance of falling on it due to imprecise control or previous momentum. Finally, in the case where no activation is present on the landscape around the agent, i.e. all directions are of equal saliency, agents engage in a form of random walk reminiscent of similar exploratory behaviour that can be seen in many simple animals. The temptation to interpret these actions in a human or animal context is ever present–agents can seem to exhibit surprise on encountering an unexpected danger, confusion if trapped in a mediocre part of the landscape and even happiness as they gallop toward the resource. The reader is encouraged to view example behaviours by watching the video at http://eprints.keele.ac.uk/rt4eprints/file/2093/.

Quantitative Results. The fitness scores of the three agents in each tournament were collected. Figure 5 shows the progress of the population from a typical run, in solving each level of the 3D RC task. Table 1 shows an overview of the performance of the entire system by aggregating and examining the results of the final 1000 tournaments from each run. From this table, it can be seen that every run was able to complete levels one and two in at least 80% of the final 1000 tournaments, and 95% of runs were able to complete level three to this standard too. Performance fell sharply against the bridge-building challenges, although 10% of runs were still able to complete level four in at least 80% of evaluations. At the hardest level of the task, 65% of runs achieved at least 1 evaluation which was able to complete level 6, and 20% of runs achieved at least 20% evaluations able to complete level 6. Figure 6 shows this aggregate data for all runs and levels and makes clear the spread of success across the whole problem in the experiment; a clear divide can be seen between the first half and latter half of the problem.

When examining the progression of the evolutionary algorithm in individual runs, it can be seen that the first level of the problem is solved early on in the search–typically after only 10000 tournaments. Success at level two soon follows as the problems are similar. Success at the third level (traps and stones, but no river) also occurs early on, in most runs. Levels four, five and six cause a longer delay in the search, and solutions do not appear at all in some runs even though the earlier levels have been solved in similar time to other, successful runs. When solutions do occur, there is often a delay between the solution for level four and later levels.

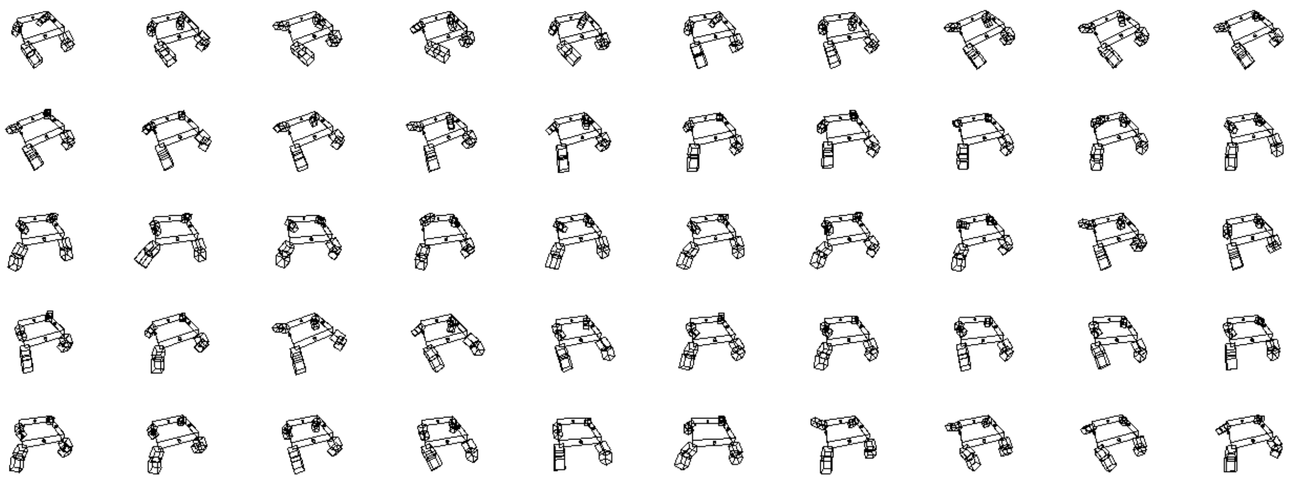


Figure 3: Example of a “galloping” locomotive behaviour. Time axis is left to right, top to bottom.

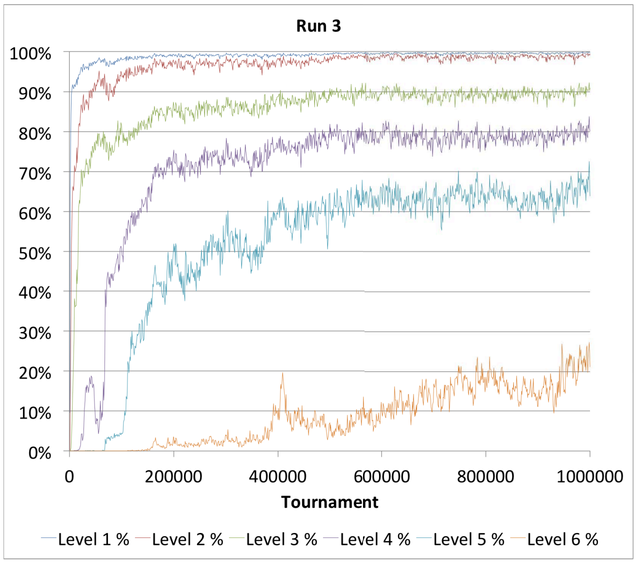


Figure 5: Progress of a typical run over one million tournaments. The graph shows the percentage of evaluations successful at completing each level of the 3D RC task, averaged over 1000 tournaments.

Cover Level	>0%	20%	40%	60%	80%
1 (Food)	100%	100%	100%	100%	100%
2 (Dash)	100%	100%	100%	100%	100%
3 (Traps)	100%	100%	100%	100%	95%
4 (River 1)	85%	85%	85%	30%	10%
5 (River 2)	85%	65%	50%	20%	-
6 (River 4)	65%	20%	-	-	-

Table 1: Proportion of runs with >0%/20%/40%/60%/80% of their final 1000 tournaments successful at level 1/2/3/4/5/6 of the 3D RC task.

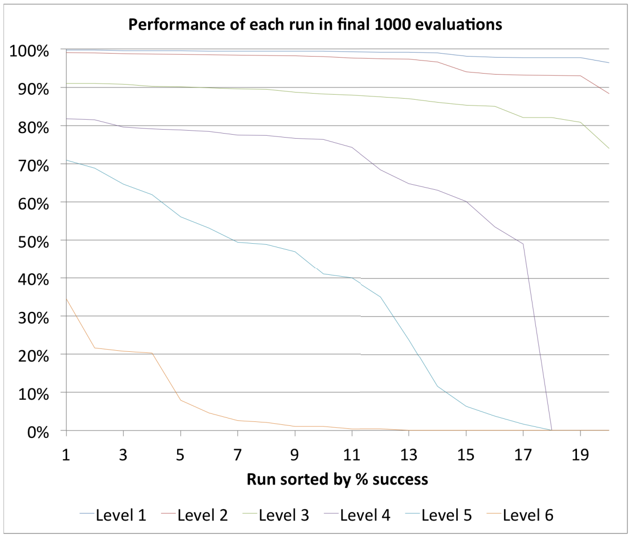


Figure 6: Success rates of all runs. The graph shows the performance of each 1000000-tournament run, evaluated from the final 1000 tournaments (3000 evaluations) of each run as the number of these evaluations that successfully completed each level of the 3D RC task. Runs are sorted in descending order for each level of the task.

Conclusions and Future Work

This work demonstrates that a standard evolutionary algorithm is sufficient to find parameters for a hybrid neural architecture comprised of loosely-coupled continuous-time and discrete-time neurons to produce reactive and deliberative behaviour in 3D, rigid-body virtual creatures requiring motion control.

By covering the range of task complexity over evolutionary time, species experience an evolutionary pressure (no loss of gradient) whilst still being able to consolidate progress already made. This incremental approach allows species to first develop a locomotive behaviour, and then to use and adapt this ability to explore the space of solutions to the bridge-building river-crossing task.

This work has also shown that a hybrid approach to neurocontroller design that includes a generalised oscillatory component (in this case, an evolved network of leaky integrators) is sufficient to produce agents that exhibit task-dependent behaviours including locomotion, turning and avoidance. The architecture is also able to optimise the strategy for long-term deliberative planning in the 3D RC world at the same time.

The integration of a deliberative decision network and a mechanism to generate reactive behaviour in 3D virtual creatures, via a shunting landscape model, was successful and shows promise for future, more complex work in this area. The limitations of the model are due to the simplicity of the decomposition of the world into the agents' phenomenal space–there is no reason this relationship could not be integrated.

In order to generalise the applicability of this work to a broad range of tasks, it will be necessary to remove the problem-specific aspects of the neural architecture's design. A first step could be to make the distinction between the DN, SM and PN less explicit. Ultimately a single neural type and architecture, with genetically specified parameters, would be the most general design.

Other possibilities for increasing the coherence in the sensorimotor loop include finer-grained distinctions in the environment, for example iota values for boundary conditions, and the addition of noise to smooth behavioural transitions.

Associated Content

A video showing an agent completing a full run of tests is available at: http://eprints.keele.ac.uk/rt4eprints/file/2093/

References

Beer, R. D. and Gallagher, J. C. (1992). Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, 1(1):91–122.

Borg, J. M., Channon, A., and Day, C. (2011). Discovering and maintaining behaviours inaccessible to incremental genetic evolution through transcription errors and cultural transmission. In *Advances in Artificial Life: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems (ECAL 2011)*, pages 101–108.

Channon, A. D. and Damper, R. I. (1998). Evolving novel behaviors via natural selection. In *Proceedings of Artificial Life VI*, pages 384–388.

Cully, A. and Mouret, J.-B. (2015). Evolving a behavioral repertoire for a walking robot. *Evolutionary Computation*.

Glasius, R., Komoda, A., and Gielen, S. C. (1995). Neural network dynamics for path planning and obstacle avoidance. *Neural Networks*, 8(1):125–133.

Grossberg, S. (1988). Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural networks*, 1(1):17–61.

Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *Physiology*, 117(4):500–544.

Langton, C. G. (1989). *Artificial Life: Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems*. Addison-Wesley Longman Publishing Co., Inc.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of Artificial Life XI*, pages 329–336.

Lessin, D., Fussell, D., and Miikkulainen, R. (2013). Open-ended behavioral complexity for evolved virtual creatures. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pages 335–342.

Luo, C., Gao, J., Li, X., Mo, H., and Jiang, Q. (2014). Sensor-based autonomous robot navigation under unknown environments with grid map representation. In *Swarm Intelligence (SIS), 2014 IEEE Symposium on*, pages 1–7.

Meng, M. and Yang, X. (1998). A neural network approach to real-time trajectory generation [mobile robots]. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 2, pages 1725–1730.

Miconi, T. and Channon, A. (2006). An improved system for artificial creatures evolution. In *Proceedings of Artificial Life X*, pages 255–261.

Pfeifer, R., Iida, F., and Lungarella, M. (2014). Cognition from the bottom up: on biological inspiration, body morphology, and soft materials. *Trends in Cognitive Sciences*, 18(8):404 – 413.

Reil, T. and Husbands, P. (2002). Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation*, 6(2):159–168.

Robinson, E., Ellis, T., and Channon, A. (2007). Neuroevolution of agents capable of reactive and deliberative behaviours in novel and dynamic environments. In *Advances in Artificial Life: Proceedings of the Ninth European Conference on the Synthesis and Simulation of Living Systems (ECAL 2007)*, pages 345–354.

Rosch, E., Thompson, E., and Varela, F. J. (1991). *The embodied mind: Cognitive science and human experience*.

Sims, K. (1994). Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 15–22.

Stanton, A. and Channon, A. (2013). Heterogeneous complexification strategies robustly outperform homogeneous strategies for incremental evolution. In *Advances in Artificial Life: Proceedings of the Twelfth European Conference on the Synthesis and Simulation of Living Systems (ECAL 2013)*, pages 973–980.

Yaeger, L. (1993). Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or PolyWorld: Life in a new context. In *Proceedings of Artificial Life III*, pages 263–298.

Achieving Compositional Language in a Population of Iterated Learners

Lewys Brace*, Seth Bullock, and Jason Noble

Institute for Complex Systems Simulation, University of Southampton, UK.

*L.G.Brace@soton.ac.uk

Abstract

Iterated learning takes place when the input into a particular individual's learning process is itself the output of another individual's learning process. This is an important feature to capture when investigating human language change, or the dynamics of culturally learned behaviours in general. Over the last fifteen years, the Iterated Learning Model (ILM) has been used to shed light on how the population-level characteristics of learned communication arise. However, until now each iteration of the model has tended to feature a single immature language user learning from their interactions with a single mature language user. Here, the ILM is extended to include a population of immature and mature language users. We demonstrate that the structure and make-up of this population influences the dynamics of language change that occur over generational time. In particular, we show that, by increasing the number of trainers from which an agent learns, the agent in question learns a fully compositional language at a much faster rate, and with less training data. It is also shown that, so long as the number of mature agents is large enough, this finding holds even if a learner's trainers include other agents that do not yet posses full linguistic competence.

Introduction

Human language is a learned system of symbolic representation that exhibits syntactic structure. Although the communication systems of other species appear to exhibit, at least to some degree, one or more of these features, the presence of all three in human language is arguably what makes it unique (Smith, 2002b).

Furthermore, human language has a number of notable design features, such as the way in which utterances are constructed from sub-parts, such as words and parts of words, which are reused and recombined in systematic ways. Thus, the meaning of an expression is related to the meanings of its constituent parts and the way in which they are combined. This trait enables language to be expressively open-ended, and is known as *compositionality* (Brighton and Kirby, 2001; Kirby, 2002b; Smith et al., 2003).

Kirby (2007) observes that compositionality endows human language with an obvious adaptive advantage in terms of its ability to communicate novel meanings; i.e., those that have never before been expressed. Given the utility associated with the ability to construct a wide range of messages from just a few learned basic units (Kirby, 2013), it is remarkable that we do not see compositionality being used as part of a learned mapping between meanings and signals in the communication systems of other species[1].

The view that language is culturally-transmitted, and that this may have a crucial role in shaping the way in which it is formed (Smith, 2002a; Brighton et al., 2005; Christiansen and Chater, 2008) has led to a body of work arguing that compositional syntax may have arisen, not as a consequence of its utility to us, but because it better ensures the continued existence of the language itself (Kirby, 2007). This work sees the self-preservational development of language occurring as a result of a cultural-evolutionary process termed *iterated learning* (Brighton and Kirby, 2001; Kirby and Hurford, 2002; Smith et al., 2003; Kirby et al., 2008, 2014); the process whereby an individual learns their cultural behaviour from other individuals, who have themselves acquired their cultural behaviour in the same way. In other words, the input into an individual's learning process is, itself, the output of prior learning in other individuals.

Models of iterated learning and human language, then, involve an agent being presented with a set of meanings that it wishes to convey, choosing signals for each of these meanings, and then transmitting these meaning-signal pairs, or utterances, to another agent who then learns from them. This process is repeated generation after generation, and can be seen to represent how language competence and understanding can develop through observational learning (Brighton, 2002).

The distinction made within iterated learning research between the observable speech acts that fuel language learning on the one hand, and the individual's internal learned representation of a language on the other, is reminiscent of the concepts of *I-language* and *E-language* that were originally put forward by Chomsky (1986):

I-language: This is the pattern of neurons that implements

[1] Although, as Kirby (2012) observes, bee dances do display limited compositionality.

an individual language user's grammar within their mind.

E-language: This is the set of utterances that make up the spoken language.

Deacon (1997) argues that in order for language patterns to continue from one generation to the next, there is a requirement for a mapping from I-language to E-language and back again; he termed this the linguistic *bottleneck*.

The central contribution of the Iterated Learning Model (ILM) is first to have successfully idealised this process in a simplified setting that is amenable to study, and then to have demonstrated that the character of this bottleneck is crucially important to both whether or not language can be successfully passed from generation to generation and, in the situations where this transmission can be achieved successfully, show that it is also crucial to the character of the language that arises.

There have been numerous incarnations of the iterated learning model. Kirby, for example, has used versions of the ILM to look at the recursive properties of language (Kirby, 2002a) and compositionality (Brighton and Kirby, 2001). Hurford (2000) explores generalised phrase structure, while Brighton (2002) uses an ILM to explore the concept of the poverty of the stimulus (the fact that the data available to a language learner is sparse, yet the knowledge of language that they achieve is complex) and its relationship with a genetically coded innate language acquisition device.

Support for the ILM and the role of learner bias in language change has come about in recent years from both iterated learning experiments involving human participants, which have supported much of the work previously done with computational simulation (Kalish et al., 2007; Kirby et al., 2008), and from other methods of research such as the statistical analysis work of Lupyan and Dale (2010), who found that languages that are spoken by larger groups of individuals, such as modern English, tend to have simpler inflectional morphology[2] than those spoken by smaller groups.

It has even been suggested that the rarity of language in nature could, in part, be due to the rarity of iterated learning in the natural world (Kirby et al., 2014)[3].

However, language learning in humans takes place within a complex social setting. Rather than each immature language user being assigned a single mature language user as a tutor, language users are exposed to linguistic input from a range of language users, some more mature than others.

This paper explores the changes to the behaviour exhibited by the ILM that result from situating language learning within a population of mature and immature learners. In the next section, we introduce an existing variant of the ILM, and replicate its basic findings. We then describe an extended model featuring a population of learners and present results from this model. Finally we discuss the findings and conclude the paper.

The Iterated Learning Model

There have been several published variants of the Iterated Learning Model (ILM). Here we will extend one that was originally discussed by Kirby and Hurford (2002). It has four components:

1. A finite meaning space, $\mathcal{M}$
2. A finite signal space, $\mathcal{S}$
3. One speaker
4. One learner

Here, a language is defined as a mapping between a finite space of signals and a finite space of meanings. Each meaning and each signal are represented as an 8-bit binary string:

$$\mathcal{M} = \{m_1, m_2, \ldots, m_{256}\}$$

$$\mathcal{S} = \{s_1, s_2, \ldots, s_{256}\}$$

Each agent's personal mapping from signals to meanings is implemented in the form of a three-layer feed-forward artificial neural network with eight nodes in each layer (see figure 1). Each of the eight nodes in the input layer is influenced by one of eight bits in an uttered signal. The degree of activation of each node in the input and hidden layers influences every node in the immediately downstream layer via a weighted connection. Each node's activation is determined by the weighted input it receives from upstream nodes, squashed by a standard logistic activation function:

$$y_i = \frac{1.0}{1.0 + e^{-x_i}} + \theta_i + I_i$$

$$x_i = \sum_j \omega_{ji} y_j$$

Where y_i is the activation level of neuron, i, and x_i is incoming stimulation received by i, calculated as the weighted sum of upstream activation values. Each neuron also receives a constant bias input, $\theta_i = 1.0$, and may receive an external input $I_i \in \{0, 1\}$ if it is part of the input layer.

The activation values of the output layer are then translated into an 8-bit binary meaning by thresholding each node's activation around the value 0.5. This string represents an agent's best guess as to the meaning of the utterance that was input into the network. During learning, an agent updates the weights of its network using back propagation with a learning rate of 0.1 and no momentum term (Rumelhart et al., 1986).

[2] The process whereby adding a morpheme to a word either creates a different form of the word (i.e. car ⟶ cars) or a new word with a different meaning (i.e. car ⟶ caring).

[3] It is noteworthy, that certain species of songbird appear to learn their songs through a process akin to iterated learning.

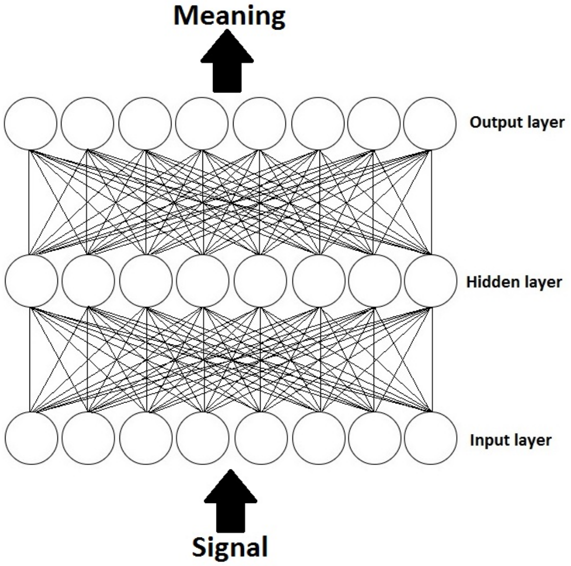


Figure 1: The agent's neural network architecture.

Initially two agents are created, a mature language user (sometimes referred to as the "speaker") and an immature language user (sometimes referred to as the "learner"). At the outset of the simulation there is no established language in place so, following Kirby and Hurford (2002), the mature language user is assigned a language comprising of a random mapping from each meaning to a randomly chosen signal. The immature language user is assigned a random neural network, i.e, each network weight is drawn from a normal distribution with zero mean and standard deviation 0.1, and each node's bias input is 1.0.

The mature language user, M, then trains the immature language user, I, for a number of training episodes, T. Each episode involves M being assigned a meaning to express and generating an associated utterance, and I using their neural network to infer a meaning associated with that utterance. Any difference between the true meaning that M attempted to express and the meaning that I infers results in back propagation making changes to I's neural network in an effort to minimise this comprehension error. Note that in order for this supervised learning to take place, ILM models assume that I is able to make use of knowledge of the true meaning that M intended to convey.

The full set of training episodes that an immature language user experiences often comprises multiple exposures to the same fixed set of unique meanings. An agent might experience E epochs of training with each epoch comprising the same set of B randomly chosen unique meanings experienced in an order that is randomised for each epoch, i.e., $T = E \times B$. The number of different meanings communicated to a language learner, B, is referred to as the language learning "bottleneck".

After all training episodes are complete, the mature language user is discarded, the immature language user is promoted to become the new mature language user, and a new randomly configured immature user is created to be trained. This process repeats for some fixed number of generations. Note that at the start of every generation the immature language user is assigned an entirely random neural network; there is no inheritance of language other than through experience of language learning episodes. Note also that the population structure is $1 + 1$. At any moment in time one mature speaker is training one immature learner.

Since ILM agents have a neural network that maps *uni-directionally* from signals to meanings, they require an additional mechanism in order to generate signals for particular meanings. To this end, Kirby and Hurford (2002) adopt the *obverter* learning procedure that was originally formulated by Oliphant and Batali (1997). Here, each speaker assumes that the hearer's internal mapping between signals and meanings is similar to its own and, consequently, when choosing which signal to make for a particular meaning, will choose the signal that, if presented as input to their own neural network, would most strongly cause them to infer this meaning, themselves. Oliphant and Batali (1997) prove that individuals using the obverter will tend to improve their communicative accuracy over time until an optimal communication system is achieved. Since the space of signals is finite and relatively small, this type of mechanism is feasible in the model.

In order to apply the obverter procedure within the ILM, Kirby and Hurford (2002) employ a confidence measure to determine which signal to produce for a given meaning. A speaker aiming to express a particular meaning, m, identifies their favoured signal, s^*, in the following manner:

For each signal, $s \in \mathcal{S}$, the speaker calculates an associated confidence value:

$$V_s = \prod_i (1 - |m[i] - o[i]|)$$

where $m[i]$ is the i^{th} bit of the target meaning and $o[i]$ is the i^{th} real valued output of the signaller's neural network. The signaller then picks s^* as the signal with the largest confidence

ILM Results

We employ three metrics to evaluate language development, expressivity, stability and compositionality. A language's *expressivity*, X, is the proportion of possible meanings that are generated by the full set of possible signals. A language with maximal expressivity is said to be complete. A language's *stability*, S, is a relational property involving two agents and is measured as the proportion of the meaning space that can be recovered accurately when one agent signals to another. When a language is maximally stable, any meaning expressed by one agent can be inferred correctly by the other.

The *compositionality*, C, of an agent's language is the extent to which utterance parts convey distinct meanings. A language with zero compositionality is one in which every utterance is paired with a meaning in an uncorrelated fashion. Knowing part of the utterance provides no knowledge of part of the meaning. A fully compositional language is one in which every part of an utterance conveys perfectly an associated part of the meaning.

We evaluate the degree of compositionality in an agent's language by first employing the obverter procedure to generate a signal for each of the meanings in the meaning space. We then calculate the values of each of the 8×8 correlations, C_{ij}, between the 256 values at the i^{th} bit of the set of signals and the 256 values at the j^{th} bit of the set of meanings. For each row, i, of this matrix we then calculate $C_{i*} = \max_i C_{ij}$, the maximum correlation between the values at index i of the signal set and the values at each of the indices of the meaning set. Finally, compositionality, C is calculated as the average of these eight maximal correlation values, $C = \frac{1}{8}\sum_i C_{i*}$. For a random language mapping meanings to signals, $C = 0.5$. Where a complete language is fully compositional, $C = 1$, each bit in an utterance conveys the value of one bit in the associated meaning.

The model displays three different types of behaviour, depending upon the size of the bottleneck. If the bottleneck is too small, then the agents do not learn; this results in a language that is both inexpressive and unstable. If, however, the bottleneck is too big, then an expressive and stable system is eventually reached; although, only after a prolonged period of time. Agents quickly achieve a language that is expressive and stable (see figure 2) and fully compositional (see figure 3) with a bottleneck of size 50.

Population-based Iterated Learning

The authors of the ILM themselves point out that complex population dynamics were traded off for computational power in the original model. Population structure was not taken into account, and every agent only ever learns from one other agent[4]. Given that the iterated learning model aims to shed light on the relationship between the properties of individuals and the population-level behaviour that they exhibit, and that much of the work done in this area thus far has been concerned primarily with vertical cultural transmission, it is of significant interest to explore the behaviour of this ILM within a population of agents.

Here we introduce a model in which, at each iteration, a population of N language users comprises N_M mature individuals and N_I immature individuals, where $N_M + N_I = N$. During each iteration of the model, every immature language user is assigned a number of trainers from whom they infer the structure of their language through a series of training episodes. This set of trainers may involve both a number, M_T, of randomly chosen mature trainers, and also, possibly, a number, I_T, of randomly chosen immature trainers (see figure 4). The presence of immature trainers represents scenarios in which language learners are not kept isolated from one another, but may influence each others' language learning. An immature individual's total number of training episodes, T, is evenly split between their trainers with each trainer being involved in $\frac{B}{M_T + I_T}$ episodes per training epoch[5]. As in the original ILM, it remains the case that

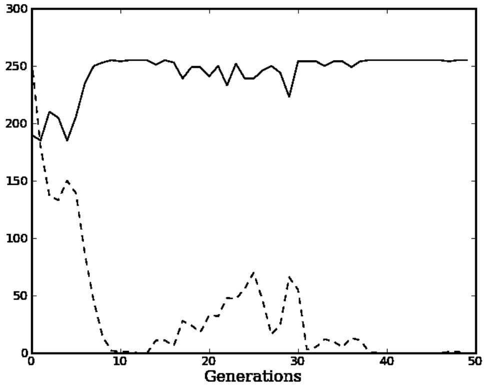


Figure 2: Replication of ILM behaviour. The solid line represents language expressivity, X, the proportion of the meaning space that is covered by the learner's language. The dotted line represents the language instability, $256 - S$, the difference between the language mappings of the mature and immature language users. Here, N_M=1, N_I=1, B=50, E=100, M_T=1, I_T=0.

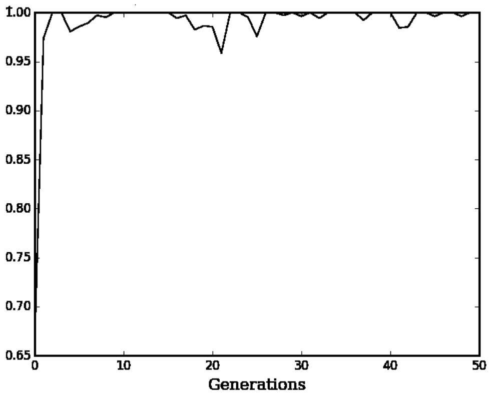


Figure 3: Langauge compositionality, C, over time for the ILM replication, where N_M=1, N_I=1, B=50, E=100, M_T=1, I_T=0.

[4] This is more than likely due to the computational capacities of the hardware at the time when the original model was developed.

[5] Fractional numbers of training episodes are avoided by round-

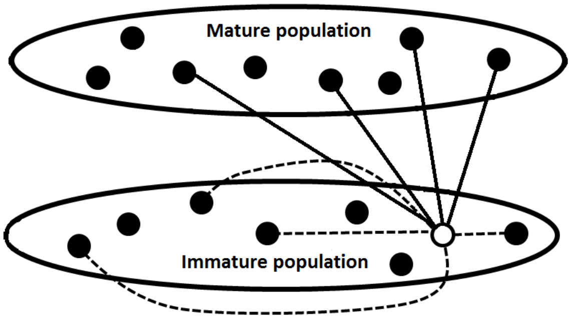


Figure 4: Diagrammatic representation of an ILM population divided into mature (upper set) and immature (lower set) agents, with $N = N_M + N_I = 16$ agents per generation. Lines represent one immature agent's trainers: four mature trainers ($M_T = 4$, solid lines) and four immature trainers ($I_T = 4$, dashed lines).

the total number of training episodes, T is the product of the bottleneck size, B, and the number of training epochs, E. Hence, $T = B \times E$. The training episodes involving a specific trainer will involve the same set of randomly selected meanings in each training epoch. The set of B training episodes that comprise a single epoch are encountered in random order.

Iterated Learning Population Model Results

Figure 5 depicts a cross section of possible combinations of M_T and I_T, and how expressivity, X, and stability, S, develops in the population model.

In comparing figure 5A with figure 2, it is clear that a training input from multiple mature agents has a significant impact upon the number of generations required for a fully expressive and stable communication system to arise. Unsurprisingly, given the nature of iterated learning, figure 2B shows how the system fails to improve above the scores obtained by random chance when M_T=0. Figures 2C and 2D depict how the system is able to produce a largely expressive and stable system when both M_T and I_T are set equal, at 5 and 10, respectively.

To further explore the impact of multiple mature trainers on model behaviour, a series of tests were conducted with the aim of exploring the linguistic bottleneck. In figure 6, we see the result that different bottleneck sizes have upon compositionality in a population where I_T=0 and E=50; meaning that agents get half of the training sessions that they did in the original model, which should make learning far more difficult. The left graph showing $M_T = 1$ and the right showing $M_T = 10$. In both graphs, it can be seen that, when the bottleneck is set too low, the system does not learn. When agents learn from only one mature trainer, a bottleneck of at least 80 meanings is required before fully compositional language can survive. However, with ten mature trainers, a high level of compositionality can arise and survive with a much smaller bottleneck of around 50. Moreover, when compositional language arises, it does so far faster when multiple trainers are present.

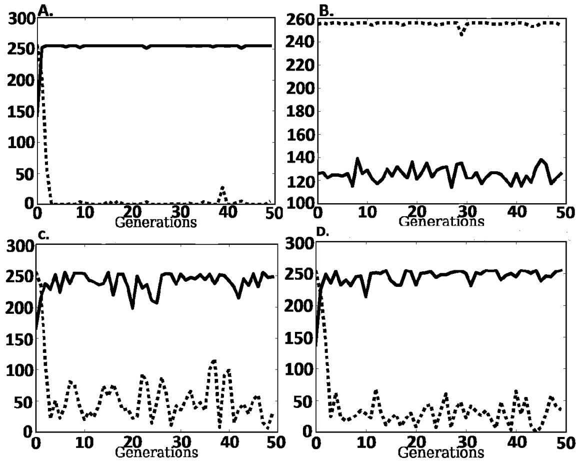


Figure 5: System behaviour for a single run of the ILPM simulation for various combinations of M_T and I_T. As above, the solid line depicts expressivity, X, and the dotted line represents instability, $256 - S$. Parameter settings are as follows: A. M_T = 10, I_T = 0; B. M_T = 0, I_T =10; C. M_T = 5, I_T =5; D. M_T = 10, I_T = 10; where N_M=15, N_I=15, B=50, and E=100 for all. Both the expressivity score and stability score are the average of the immature population after language learning has been completed.

Figure 7 depicts analogous results for scenarios in which immature language users are allowed to influence each others' learning ($I_T = 5$). When immature trainers outnumber mature trainers (figure 7 left), language learning is compromised, with compositionality varying erratically over successive generations. Despite this, it is notable that bottleneck size does influence language with larger bottlenecks allowing languages to achieve somewhat higher compositionality. When immature trainers are outnumbered by mature trainers (figure 7 right), language learning is successful for scenarios with larger bottleneck sizes, although compositionality does vary more from generation to generation by comparison with an equivalent scenario without immature trainers (compare figure 6 right).

Further evidence of M_T impacting the system behaviour can be seen in figure 8, which plots the average level of compositionality that the system exhibits per generation for various combinations of M_T and I_T. In line with the above results, it can be seen that compositional language tends to arise to the extent that the number of mature trainers is greater than the number of immature trainers, and that a greater number of mature trainers enables the system to develop and maintain a higher level of language composition-

ing up.

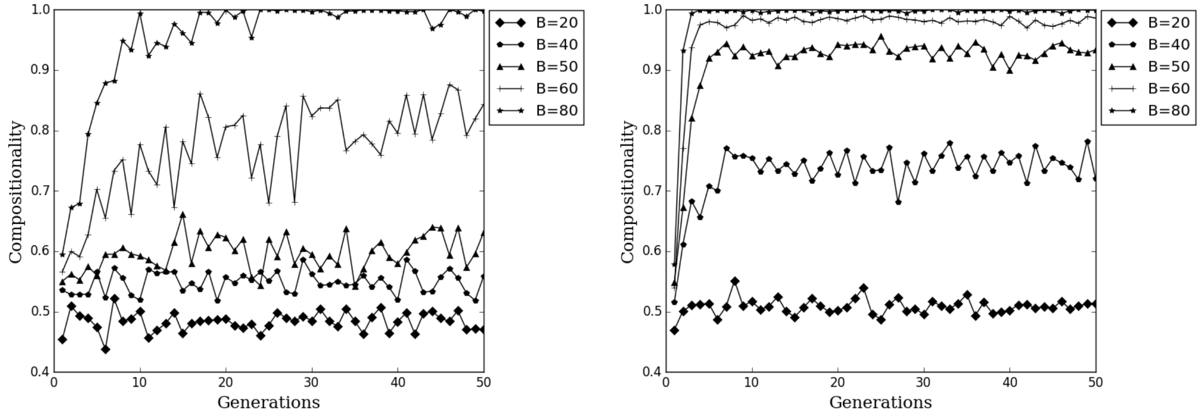


Figure 6: Graph depicting the impact of various value of B. *Left*: M_T = 1; *Right*: M_T = 10. (N_M=15, N_I=15, I_T = 0 and E=50 in both cases). The compositionality score is the average of the immature population after language learning has been completed.

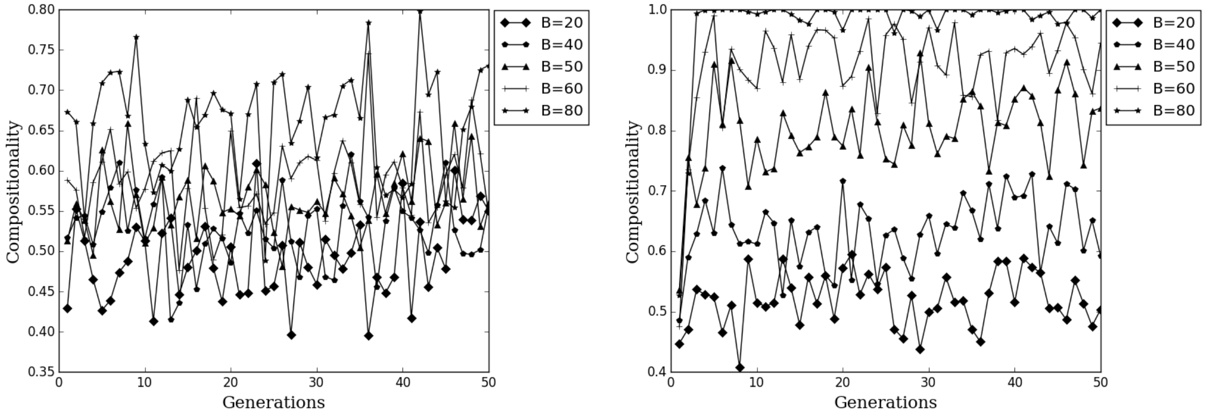


Figure 7: Graph depicting the impact of various values of B. *Left*: M_T = 1; *Right*: M_T = 10. (N_M=15, N_I=15, I_T = 5 and E=50 in both cases). The compositionality score is the average of the immature population after language learning has been completed.

ality.

Why might dividing the same number of learning episodes between a greater number of mature trainers lead to improved language learning in an immature language user? Several possibilities present themselves: multiple trainers could allow effective languages to spread through the population more quickly since one trainer can influence several learners, or, equivalently, expose learners to a sample of multiple languages, some of which may be more easy to learn and use. However, manipulating the population structure in ways that would be expected to influence this effect made no difference to performance.

Alternatively, might multiple trainers provide learners with increased diversity of language experience at the outset of the simulation, when naive neural networks tend to map many meanings onto the same signal. Figure 9 lends some support for this hypothesis, showing that the number of unique signals experienced by a language learner at generation 2 of a run is increased when the learner is exposed to multiple trainers and that this increase in diversity is directly proportional to the increase in compositionality of the language exhibited a few generations later. However, it should be noted that although there was a strong relationship between signal diversity and compositionality across scenarios that differed in terms of the number of trainers, when the number of trainers was held constant there was not always a strong relationship of this kind, suggesting that signal diversity may not be the whole story.

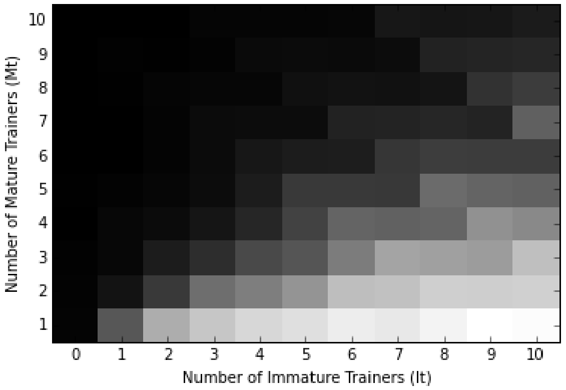


Figure 8: Heatmap of the average amount of compositionality over 50 generations, where N_M=15, N_I=15, B=50, and E=100, throughout. The compositionality score is the average of the immature population after language learning has been completed.

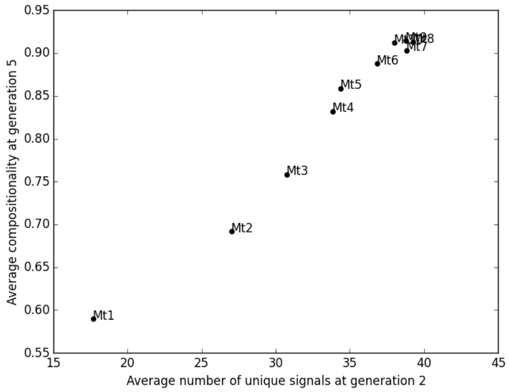


Figure 9: The relationship between the average number of unique signals that an immature learner experiences at generation 2 and the average compositionality of the language learned at generation 5 for runs with different numbers of mature trainers (M_T). Each data point represents an average over 10 runs where $N_M = 15$, $N_I = 15$, $B = 50$, $E = 50$.

Discussion and Conclusions

We have demonstrated that Kirby and Hurford's (2002) iterated learning model variant can operate successfully within a population of agents. Given an appropriately sized language-learning bottleneck, when each member of a population of immature language users learn their language from enough mature language users, the population is readily able to converge on a complete, compositional language. Moreover, increasing the number of mature trainers tends to allow compositional language to pass through a smaller learning bottleneck and to establish itself in a smaller number of generations.

Work within the iterated learning paradigm typically holds that the way in which a language changes over time can be seen as a compromise the influence of learner biases and the influence of constraints acting upon language during transmission (Kirby, 2002a; Brighton et al., 2005; Smith, 2009). For example, the transmission bottleneck favours languages that can be inferred by language learners from a limited number of utterances (Kirby, 2002a; Brighton, 2002; Smith, 2009). Thus, the compositionality of a language represents an adaptation in response to selection pressures imposed by the environment in which it must survive. It is important to understand the dynamics of iterated learning within linguistic populations since population structure may be an additional source of of constraints on language transmission and may therefore influence the form that languages tend to take over time.

The role of such constraints has been modelled previously in an iterated learning context. Griffiths (2007), for instance, explored iterated learning dynamics within a model where learning algorithms were based on the principles of Bayesian inference. By extending his framework to a population of such Bayesian agents where each learner learns from a single member of the previous generation, he showed that iterated learning in this population of Bayesian agents produced language outcomes that could be understood as solely the result of the agent's individual learning biases. Therefore negating the role of other constraints, such as the transmission bottleneck.

However, Smith (2009) argues that Griffiths' (2007) findings imply that it is possible to understand the prior biases of learners by looking at the typological distributions of languages. Smith (2009) also presents a model of Bayesian agents and demonstrates that Griffiths' results are based upon the idealisation that a learner learns from a single teacher, and once multiple teachers are included, the mapping from the learner biases to typology breaks down. Based upon this result, Smith (2009) concludes that inferring learning bias from typology could yield unsafe results. Furthermore, Griffiths' (2007) model is limited by the fact that the agents use very specific statistical learning algorithms, and are therefore not applicable to cases where the subjects of study use more general-purpose learning algorithms, which are more akin to the general purpose cognitive architecture that is likely to underpin human language (Hurford, 2014).

In a later work, Burkett and Griffiths (2010) explored the problems raised by Smith (2009) by developing a model where Bayesian agents were allowed to learn multiple languages. In doing so, they demonstrated that, so long as an agents hypothesis space explicitly takes into account the possibility of receiving input from multiple speakers with potentially different languages, then Bayesian learning does tend to reflect the learners inductive biases in the same manner as the single teacher model presented in Griffiths (2007). However, this model still makes the simplifying assumption that agents only receive input from vertical transmission; this is clearly not the case for real-life language learners, who are likely to learn from their immature peers as well as from their mature role-models.

The model presented in this paper differs in this respect in that it explores the impact of immature language users upon the learning process, and the emergence of compositionality in particular. Furthermore, unlike Burkett and Griffiths (2010), we have explored iterated learning dynamics within a population of agents who are attempting to learn a single language. We have shown here that the introduction of horizontal language transmission amongst immature language learners does not tend to prevent languages from arising if each language learner is exposed to enough mature trainers.

Acknowledgements

The authors would like to thank Simon Kirby for his useful and friendly input. This work was supported by an EPSRC Doctoral Training Centre grant (EP/G03690x/1).

References

Brighton, H. (2002). Compositional syntax from cultural transmission. *Artificial Life*, 8(1):25–54.

Brighton, H. and Kirby, S. (2001). The survival of the smallest: Stability conditions for the cultural evolution of compositional language. In Kelemen, J. and Sosík, P., editors, *ECAL 2001 - Full TITLE HERE*, pages 592–601. Springer-Verlag.

Brighton, H., Kirby, S., and Smith, K. (2005). Cultural selection for learnability: Three principles underlying the view that language adapts to be learnable. In *Language Origins: Perspectives on Evoultion*. Oxford University Press, Oxford.

Burkett, D. and Griffiths, T. L. (2010). Iterated learning of multiple language from multiple teachers. In *The Evolution of Language: Proceedings of the Eighth International Conference*, pages 58–65. World Scientific, Singapore.

Chomsky, N. (1986). *Knowledge of Language: Its Nature, Origin, and Use*. Praeger Publishers, Westport, USA.

Christiansen, M. H. and Chater, N. (2008). Language as shaped by the brain. *Behavioural and Brain Sciences*, 31:489–509.

Deacon, T. W. (1997). *The Symbolic Species: The Co-evolution of Language and the Brain.* Norton, New York, USA.

Griffiths, T. (2007). Language evolution by iterated learning with bayesian agents. *Cognitive Science*, 31:441–480.

Hurford, J. R. (2000). Social transmission favours linguistic generalisation. In *The Evolutionary Emergence of Language: Social Function and the Origins of Linguistic Form*. Cambridge University Press., Cambridge.

Hurford, J. R. (2014). What is wrong, and what is right, about current theories of language, in the light of evolution? *Humana Mente Journal of Philosophical Studies*, 27:123–133.

Kalish, M., Griffiths, T. L., and Lewandowsky, S. (2007). Iterated learning : Intergenerational knowledge transmission reveals inductive biases. *Psychonomic Bulletin and Review*, 14(2):288–294.

Kirby, S. (2002a). Learning, bottlenecks and the evolution of recursive syntax. In Briscoe, T., editor, *Linguistic Evolution through Language Acquisition: Formal and Computational Models*, pages 173–204. Cambridge University Press., Cambridge.

Kirby, S. (2002b). Natural language from artificial life. *Artificial Life*, 8(2):185–215.

Kirby, S. (2007). The evolution of meaning-space structure through iterated learning. In Lyon, C., Nehaniv, Chrystopher, L., and Cangelosi, A., editors, *Emergence of Communication and Language*, pages 253–267, London. Springer-Verlag.

Kirby, S. (2012). Language is an adaptive system: The role of cultural evolution in the origins of language. In *The Oxford Handbook of Language Evolution*. Oxford University Press, Oxford.

Kirby, S. (2013). Language, culture, and computation: An adaptive systems approach to biolinguistics. In *The Cambridge Handbook of Biolinguistics*. Cambridge University Press, Cambridge.

Kirby, S., Cornish, H., and Smith, K. (2008). Cumulative cultural evolution in the laboratory: An experimental approach to the origins of structure in human language. *Proceedings of the National Academy of Sciences of the United States of America*, 105(31):10681–6.

Kirby, S., Griffiths, T., and Smith, K. (2014). Iterated learning and the evolution of language. *Current Opinion in Neurobiology*, 28:108–114.

Kirby, S. and Hurford, J. (2002). The emergence of linguistic structure an overview of the iterated learning model. In Cangelosi, A. and Parisi, D., editors, *Simulating the Evolution of Language*. Springer, London.

Lupyan, G. and Dale, R. (2010). Language structure is partly determined by social structure. *PLoS ONE*, 5(1):e8559.

Oliphant, M. and Batali, J. (1997). Learning and the emergence of coordinated communication. *The Newsletter of the Center for Research in Language*, 11(1):1–46.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Smith, K. (2002a). The cultural evolution of communication in a population. *Connection Science*, 14(1):65–84.

Smith, K. (2002b). Natural selection and cultural selection in the evolution of communication. *Adaptive Behavior*, 10(1):25–45.

Smith, K. (2009). Iterated learning in populations of Bayesian agents. In Taatgen, N. A. and van Rijn, H., editors, *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, pages 697–702. Cognitive Science Society, Austin, TX.

Smith, K., Kirby, S., and Brighton, H. (2003). Iterated learning: A framework for the emergence of language. *Artificial Life*, 9(4):371–86.

The Hunger Games:
Embodied agents evolving foraging strategies on the frugal-greedy spectrum

Nathanaël Aubert-Kato[1], Olaf Witkowski[2] & Takashi Ikegami[2]

[1]Ochanomizu University, Tokyo
[2]The University of Tokyo, Tokyo
aubert.kato.nathanael@ocha.ac.jp, olaf@sacral.c.u-tokyo.ac.jp

Abstract

In Evolutionary Biology and Game Theory, there is a long history of models aimed at predicting strategies adopted by agents during resource foraging. In Artificial Life, the agent-based modeling approach allowed to simulate the evolution of foraging behaviors in populations of artificial agents embodied in a simulated environment.

In this paper, different sets of behaviors are evolved from a simple setting where agents seek for food patches distributed on a two-dimensional map. While agents are not explicitly playing a game of chicken, their strategies are found on a spectrum ranging from a frugal strategy (aka *Dove*) to a greedy strategy (aka *Hawk*). This phenomenon is due to the fact that moving is both a way for the agents to play or go to get away from an unfavorable area of the environment. It is also observed that by moving away, the agents preserve the ecology, preventing the resource from disappearing locally.

Those strategies are shown to be stable if the environment is colonized by one given population. However, *post-mortem* tournaments among different groups of agents (separately evolved), systematically result in a specific group of agents dominating. The optimal strategy in the simulated tournaments is found to be one with fine-tuned timing for leaving. Further analysis shows how the strategy exploits resources without completely depleting them, producing Volterra-like population tendencies.

Introduction

In nature, animals are known to adopt a broad diversity of strategies to forage for resources (O'brien et al. 1990). While some preserve local ecology, others act more aggressively, leading to the deterioration of the local environment or ecological niche (Odling-Smee et al. 1996, Gyllenberg and Parvinen 2001, Kotler et al. 2002). This forces the group to constantly move to new sectors where the resource is plentiful, inducing a risk linked to the uncertainty related to discovery of those areas. A species may evolve a behavior that leads to its own extinction, a phenomenon called Tragedy of the Commons Hardin (1968), Matsuda and Abrams (1994).

In the field of optimal foraging theory (OFT), many models have been developed to predict the behavior of animals foraging for resource patches (MacArthur and Pianka 1966, Pyke 1984). The Ideal Free Distribution theory (Fretwell and Lucas 1970) predicts that if certain minimal conditions are fulfilled, the distribution of animals among patches will minimize resource competition and maximize fitness. The Marginal Value Theorem (Charnov 1976) describes the strategy that maximizes gain per unit of time in systems where resources decrease over time. As those models generally rely on mechanisms akin to darwinian selection to evolve the behaviors (Werner and Hall 1974), the individuals are assumed to maximize their optimal benefit per cost, in order to maximize their fitness. Different agents may change their optimal behavior in agents. This interaction has been studied extensively in the past for the case of prey-predators models (Huffaker et al. 1963, Glass 1971, Turchin 2003).

In game theory, the war of attrition (Smith 1974) is a game in which two players compete for a unitary resource and the winner is the contestant that is prepared to go on longer. In nature, there are countless examples in which individuals bid amounts of time they are ready to spend in order to get a resource, mostly for resource ownership (Smith 1982). This can also be linked to the snowdrift game (Sugden 2004), where the worst outcome is obtained when both players refuse to give up. In this game, players have to choose one of two strategies, one aggressive/greedy named *Hawk* and one frugal/cooperative named *Dove*. The greatest payoff, the temptation T is obtained by an Hawk facing a Dove. However, this comes at a risk, as the worst payoff, the punishment P is obtained by an Hawk facing another Hawk. A Dove will get a reward R when facing another Dove, and the sucker's payoff S when facing an Hawk. These payoffs are such that $T > R > S > P$, so that Dove is the safest strategy, but Hawk gives the highest possible payoff.

Applied to the problem of foraging, individuals locally play a variant of the war of attrition, in the sense that exploiting the same resource for the same time will end up being disadvantageous to the forager. This can be caused by the resource being depleted beyond a self-regeneration threshold, the resource becoming less rewarding or the foraging itself becoming more costly (Davies et al. 2012). This is similar to the law of diminishing returns (Lipsitch et al. 1995), where the animal must find out when it is more beneficial to stay or leave a resource. As such, the potential strategy are also similar to those of Hawk and Dove: one can either bet on the others leaving, which might lead to a depleted resource, or leave beforehand for a hopefully more plentiful spot.

The goal of this research is to investigate the behavior of individuals playing a continuous, spatial variant on the war of attrition or snowdrift (Sugden 2004). We are especially interested in the emergence of behaviors that are in between the expected strategies from the discrete version of the game, and the way those strategies interact, when mixed in a population.

In this paper, we thus make use of agent-based modeling (ABM) with a simplistic setup to investigate foraging behavior. We do not explicitly seek the optimal behavior, but rather interpret the behavioral data generated by modeling optimal adaptations to environmental niches (Seth 2007), and focus on analyzing the interaction of agents adopting different strategies (Stephens and Krebs 1986). Several type of behaviors are evolved, ranging over a spectrum going from frugal to greedy. We found examples of evolved behaviors from any position of the spectrum, forming their own niche. Once those behaviors are brought together, we observe however that a compromise turns out to be the best solution. We also show that this solution corresponds to one of a few populations of agents, that have fine-tuned their timing of leaving the resource patches. This “leaving” behavior with a precise timing allows to exploits resources without completely depleting them, producing Volterra-like population tendencies.

Model

We simulate a population of individuals controlled by neural networks, moving about on a two-dimensional toroidal map (Figure 1). The environment is composed of mostly empty space with a preset number of food patches randomly distributed.

The agents have to forage for food, giving them the energy they need to survive and produce offspring throughout the simulation. Agents movements and decisions are calculated per iteration, which represents the quantum of simulated time. Each iteration, the output of the neural network of an agent, as well as its position and energy are updated.

Methodology

Agent are embodied in the sense that they have a position in the environment. Overlap is allowed: it is possible for two agents to occupy the exact same spot. Agents have an internal amount of energy that is depleted over time. This energy can be increased by staying on food patches, up to a fixed maximum. If the internal energy of the agent reaches zero or below, the agent ”dies” and is removed from the simulation. Agents also die once they lived until their maximum age (1000 iterations). This approach is aimed at favoring the apparition of new genotypes and behaviors.

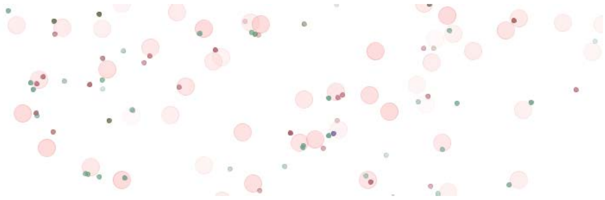


Figure 1: Simulation map. Every agent is represented by a small circle, with color representing its genotype (i.e. a vector encoding the weights of its neural network) and color intensity representing its current energy level. The large red circles represent the resource patches, where lighter color indicates that the state of depletion of the patch.

Agents take actions based on the output of their neural controller, which is implemented with an Elman artificial neural network (Elman 1990) with an architecture in three layers, similar to (Witkowski and Ikegami 2014). The network consists of two input units (encoding the current energy of the agent and the amount of energy received at this iteration), fully connected to two hidden units, themselves fully connected to two outputs units (deciding their steering angle and speed).

Agents are thus not directly playing the snowdrift game in the classic sense. Nonetheless, their position is a form of play, since they can choose how long they will remain on a food patch, or how often they will move.

All nodes in the neural network take activation values between 0.0 and 1.0. All output values are also floating values between 0.0 and 1.0, the first motor output is then converted to an angle between $-\pi$ to π, and the second motor output converted to a speed factor multiplying the velocity. The activation state of internal neurons is updated according to a sigmoid function.

The network’s weights are evolved following a similar algorithm to previous work (Witkowski and Aubert 2014): each agent, when it reaches a given energy level, produces an offspring. Each weight in the offspring’s network is mutated with the rate given in Table 1. The offspring will start with a set initial energy, equal to the energy lost by the parent. Reproduction is always asexual, with only one parent.

Energy gathering

Food patches provide energy to the agents that are on top of it. However, overfeeding leads to a depletion of a patch’s internal energy. Energy is stable if there is only one agent, decreasing if there are multiple agents, and slowly recovering if no agent is present. Each agent receives a reward equal to maximum reward per patch times the current energy fraction of the patch. Note that for a patch more that 50% depleted, agents are actually losing energy overall. It is also possible

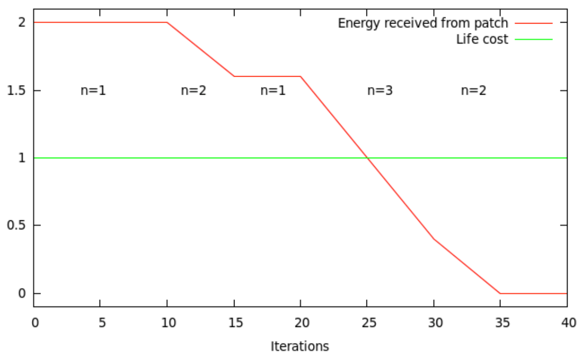


Figure 2: Typical energy intake over time for an agent on a patch. The number n of agents on the same patch fluctuates over time. When this number is higher than one, the patch gets depleted, in turn leading to a lower amount of food dispensed in total by that patch.

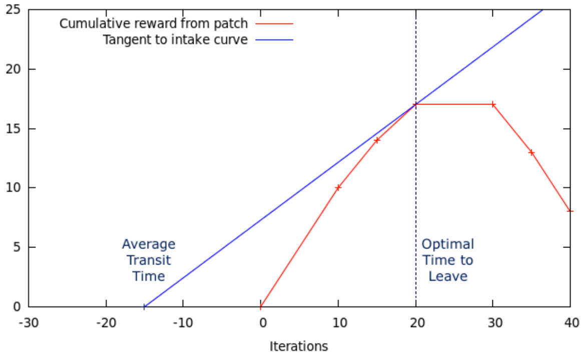


Figure 3: Cumulative resource intake and optimal time to leave a patch. This diagram illustrates the optimal point to leave a resource patch in order to maximize the amount of energy gathered per time spent foraging and traveling, by connecting the average transit time (arbitrarily 25 iterations in this example) on the horizontal time axis tangentially to the cumulative resource intake. The resulting optimal time to leave a patch is therefore 20 iterations.

for two patches to overlap, in which case agents get the sum of all rewards. As mentioned above, agents reproduce when their total energy reach the reproduction value, which can thus be considered a soft maximum.

Depletion rate per agent and recovery speed are shown in Table 1. If a food patch is completely depleted, it is destroyed and a new patch is generated at a random position. This keeps the total number of patches identical throughout the simulation.

A typical example of energy consumption over time, by an agent on a patch, is shown in Figure 2. The energy values are here arbitrary. The energy intake starts dropping when more than one individual are on the same patch. The optimal time spent on a patch, maximizing the overall ratio between resource intake and time spent foraging and traveling, can be visualized by connecting the average transit time on the x axis tangentially to the cumulative resource intake (see Figure 3). The optimum is however expected to change over time based on the interaction with other agents, which will be dependent on evolutionary dynamics.

Experimental setups

The experiment is separated in three stages. First, in the training phase, agents neural controllers are independently evolved, and the surviving agents are selected for the next phase. Second, in the analysis phase, these strategies are analyzed. Thirdly, in the tournament phase, the resulting agents are evaluated against each other.

At the beginning of the first phase, in each simulation, the world is populated with 500 random individuals. This population is evolved for 5000 iterations, which yields in most cases a uniform population (in the sense that all agents alive have a relatively close common ancestor). The last 5 generated individuals are then stored for the next phase. This approach allows us to get a sampling of the evolved strategy in the run. This was repeated over 6 runs to gather a total of 30 agents. While the sampling might be insufficient to capture completely the strategy of a given population, it was enough to gather a variety of behaviors.

During the second phase, mutations were disabled and the world was seeded with one agent at a time. Agents behavior was categorized by hand and then linked to metrics from runs. Those behaviors are detailed in the next section.

Finally, in the tournament phase, in order to evaluate all the strategies the agents evolved, we performed 100 runs seeded with the 30 individuals. Mutations were again prevented to ensure that the strategy is kept intact throughout the run. This is akin to dilating the time scale from the ”evolutionary” scale to a sort of instantaneous interaction scale. This was done in particular to protect highly tuned strategies that does not resist well to mutation.

As a complement, we also performed the same tests where agents were allowed to evolve (i.e. where mutation was active, allowing for progressive change in the behavior of newborn agents with respect to their parents’), which is similar to introducing all species in a common environment. In this case, the robustness of the strategy and its potential adaptability was the paramount factor, yielding slightly different results.

Results

In simulations, a full spectrum of behaviors have emerged. While they all lead to agents to consuming the food resource, the level of selfishness varies, ranging anywhere between frugal (*dove*) and greedy (*hawk*) strategies. The most common approaches observed are listed below, in increasing order of greediness. The distribution of agents among those

Parameter	Value
World height	800
World width	800
Initial pop (normal run)	500
Initial pop (tournament)	30
Max population	10000
Starting energy	100
Reproduction energy trigger	200
Reproduction cost	100
Existence cost	1
Food patches	200
Patch size	20
Maximum reward per patch	2
Patch recovery per iteration	0.25
Energy decrease per agent	0.04
Maximum patch energy	50
Maximum speed	10
Maximum age	1000
Mutation rate	10%
Mutation factor	0.1

Table 1: Simulation parameters. Note that we implement a maximum population, but that, with the current settings, this limit is never reached.

strategies is shown in Table 2. Note that the Table indicates a larger variability in strategies than expected for purely uniform populations. This is due to the fact that agents close in genotype space can have different behaviors due to mutations.

Butterfly Those agents are not staying long in a given place. They tend to gather energy for a few iterations, then move to another patch in straight line. Overtime, agents tend to accumulate enough energy to make children, making this strategy viable. The overall amount of generated offspring is low, and mortality rate is high. This strategy can thus be outperformed by most other. However, in a setting where multiple aggressive strategies are competing against each other, butterflies are mostly unaffected, and can sometimes weather the fight. In this case, they remain the last species standing.

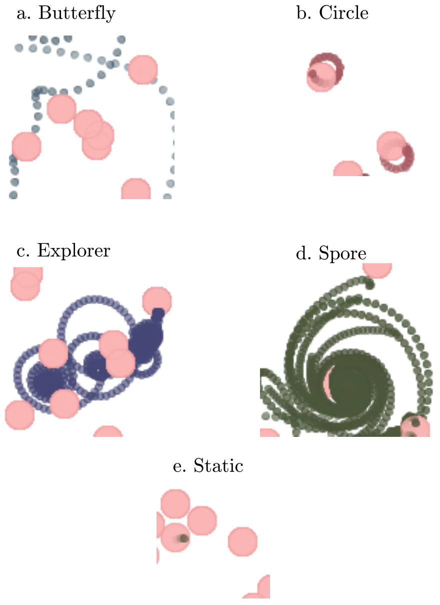
Circle Those individuals are staying close to a given food patch, but making big circles so that only a fraction of their time is spent on the food patch. This strategy keeps the total number of agent on the patch.

Explorer In this case, agents find new food patches, stay long enough to reproduce. Once the energy level of the food patch starts decreasing, they move on to the next spot. This strategy can be considered the ”average” of the spectrum.

Spore With this approach, agents colonize a patch, reproduce until near exhaustion, and then massively spread at once when the remaining energy is not enough to offset the leaving cost. This strategy yields ”bursting” events, similar to the release of spores or viruses. These spores then populate the nearby food patches and repeat the process.

Static While this is not a viable strategy for a whole species, or even for a group, a few agents evolved this behavior. Those agents typically belong to species with behaviors on the Hawk side of the spectrum, that is, behaviors that favor staying on food patch as long as possible. Pushed to the extreme, such strategies will prevent agents from moving even once the spot has been completely depleted, which leads to the agents’ death. However, during their lives, they may produce offspring with a viable strategy, closer to that of the rest of the species they belong to.

Additionally, it is possible that they will prevent the invasion of a species with a more frugal approach. Indeed, other agents will tend to leave an area with mostly depleted food patches. In that sense, the static strategy is detrimental in a uniform population, but can arguably help ensure to an extent the survival of their species in a mixed environment. This strategy might also have an interest in our particular setting, since completely depleting a food patch will create a new, full, food patch somewhere else in the environment.



a. Butterfly

b. Circle

c. Explorer

d. Spore

e. Static

Figure 4: Various strategies implemented by agents. Note that, due to mutations, an agent can have a different strategy than the species it belongs to.

Characterizing behaviors

The previous behaviors can be characterized in two ways: looking either at the overall movement of the agents or at

Butterfly	Circle	Explorer	Spore	Static
4	5	8	7	6

Table 2: Distribution of agents from 6 runs over the various evolved strategies.

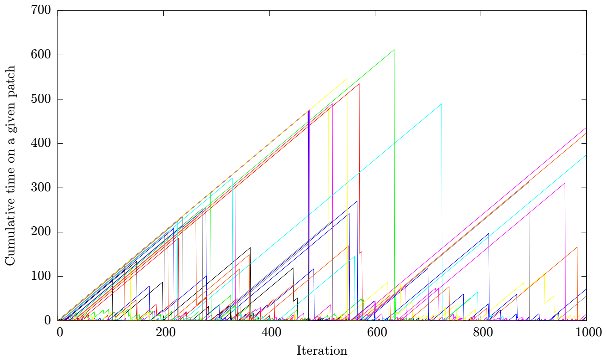


Figure 5: Cumulative stay on a given food patch for a wide variety of strategies. Frugal strategies are characterized by a succession of small peaks, while strategies on the Hawk side of the spectrum favor only one spot for extended period of time, leading to a few very large peaks.

the time they spend on food patches.

The first point can be highlighted by tracking agents over a few iterations (Figure 4). This allows us to see the path taken by agents, showing strong variations among strategies, as mentioned in their respective descriptions.

The second approach, dubbed cumulative stay analysis, was realized on a run starting with 30 independent agents. These agents were selected from the uniform population of multiple runs which evolved different strategies. We took five representatives of each of those runs and used them to seed the population. The cumulative time spent on a food patch by those agents is shown in Figure 5

Note that we did not take into account the number of offsprings generated by the different strategies. Instead we focused on the 30 sampled agents and the qualitative difference in their behaviors. Since these behaviors are dependent on being part of a population, we disabled the mutations, so that all agents ran during a specific analysis would have the same strategy.

Tournament

In a first attempt to compare strategies, we seed the world with the 30 individuals sampled from the training phase. To ensure that those strategies do not drift during the evaluation, mutations were disabled. While artificial, this approach is similar to changing the time scale: here, mutations can be considered so slow that they do not happen over the course of the evaluation. This can be seen in biology, for instance with bacteria where the time-scale of evolution of strategies and the scale of using those strategies is widely different (Kerr et al. 2002).

Individual 16	Individual 17
24	76

Table 3: Winning rates of both agents over 100 runs. No other winner was observed.

When seeded with the sampled agents, the world is quickly overrun by one of two possible agents, dubbed Individual 16 or Individual 17[1]. Since mutations are disabled, a given "species" is thus only comprised of copies of those very agents. Around 10000 iterations, only one set remains. The winning rates over 100 runs are shown in Table 3. We could not observe any other agent achieving a full population overrun, that is, reaching a state where all live agents are a copy of itself. Neither could we find stable mixed populations, even within evolutionary time much shorter than that needed for genetic drift to leave only one species in control experiments.

Figure 6 depicts the phylogeny of a typical tournament run. At the center of the plot is the root of the tree, corresponding to time zero in the simulation, with 30 initial branches. As these branches progress outward, they ramify into each agent's successive generations of offspring. The time scale is preserved, totaling 20000 iterations. Every fork corresponds to one parturition, with the newborn forking clockwise and the parent counterclockwise.

Strategy analysis To explain the overwhelming dominance of one genotype over all the others, in each simulation, we take a close look at the details of the behavior it generates. By the previous analysis, they would be categorized as "circle", although they show a behavior closer to the "explorer" strategy when the food becomes scarce. In both cases, multiple copies of the agents can be supported by a given food patch, since they are never on the patch for long and end up taking turns. Once the food is nearly depleted, they actively look for a fresher patch, giving time for the patch to regrow. It is possible to recognize this behavior in agents simply by observing the outputs of their neural network (see Figure 7). Typical inputs and outputs of Individual 17 over its life time are shown in Figure 8. Other agents are left with two choices: either stay longer, making the effort to completely destroy the patch, or leaving earlier, leaving more food for those who remain. As such, it seems that Individuals 16 and 17 have simply evolved an efficient patch finding strategy, and tuned their decision parameters to leave at the most appropriate time.

Predator-Prey oscillations of Individual 17 Since Individual 17 does not consume completely food patches, and

[1] These numbers are based on their index in the seed.

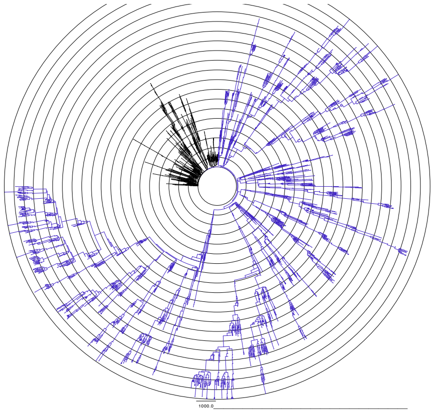


Figure 6: Phylogenetic tree of a typical tournament, won by Individual 17. The root of the tree, the inner circle at the center of the plot, corresponds to time zero in the simulation, with 30 initial branches. As these branches progress outward, they ramify into each agent's successive generations of offspring. Every circle corresponds to 1000 iterations, up to a total of 20000. Every fork corresponds to one agent replicating, one branch corresponding to the parent, and the other branch to the offspring. The subtree corresponding to the lineage still surviving at iteration 20000, descending from Individual 17, is highlighted in blue.

even ignores those that are nearly depleted, we can observe Volterra-like oscillations[2] in its population (Figure 9). The difference comes from the fact that the regrowth of the "prey" (the food patches) is linear in time, instead of the usual autocatalytic, and thus exponential, generation. Runs are spread in phase space, due to the time it takes to the population to get homogeneous. However, amplitude and frequency are roughly uniform across runs. This is in part due to the fact that Individual 17 does not destroy food patches, so that the total energy available in the system changes smoothly over time, which mitigates potential irregularities.

Tournaments with evolution In the case where agents are allowed to evolve over time (i.e. with a non-zero mutation rate on their genotypes) during the tournament, winning strategies are much more diverse. Over 100 runs, agents with a spore approach now win in a majority of cases, while circles get second best, mostly through Individual 16 and Individual 17 (Table 4). Interestingly, in one instance, a static agent was able to seed the winning population, as its offspring were able to recover a spore strategy through mutation. Note that a winning population expresses a range of behaviors, but that the original strategy remains dominant. As with the tournaments without evolution, in all runs the population is eventually overrun by a species descending from a single original agent.

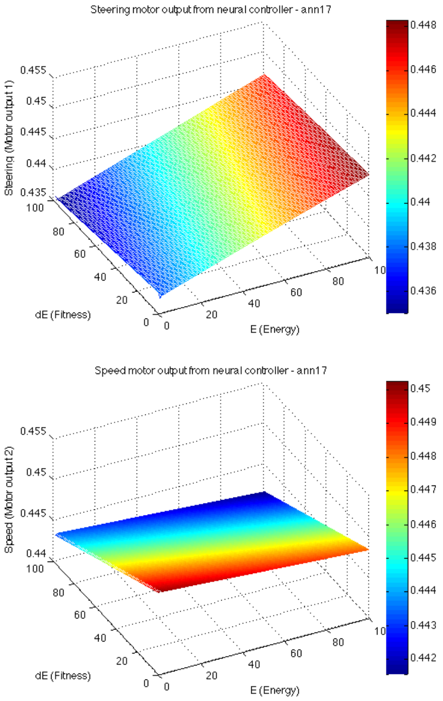


Figure 7: Motor response of Individual 17's neural controller for steering (*top plot*) and speed (*bottom plot*), versus energy and fitness inputs normalized between 0 and 100.

Strategy	Circle			Explorer			Spore			Static
Agent	16	17	19	1	8	13	6	9	15	10
Wins	11	12	6	1	4	5	30	29	10	1

Table 4: Winning counts in tournaments with evolution, separated by agents and strategies.

Those results contrast with those obtained without mutation. This may be due to aggressive agents adapting more efficiently to a competitive environment and/or to circle strategies being unstable to mutation.

Discussion

The obtained results show the emergence of several sets of behaviors from a simplistic foraging task. All agents evolve a way to find more resource, and some are also found to escape from areas when they become less beneficial. This is similar to Aktipis (2004) where agents outperform more complex strategies by simply walking away from an unfavorable location. The observed behaviors are however richer in complexity than a mere exit strategy, ranging from very static/greedy to more exploring/loose types of motion. On the greedy side, agents are exploiting the resources available to their limit, potentially exploiting the fact that new resources are then created as result. Once a food patch is removed, such agents move to the nearest available spot.

[2] Lotka (1910), Volterra (1926)

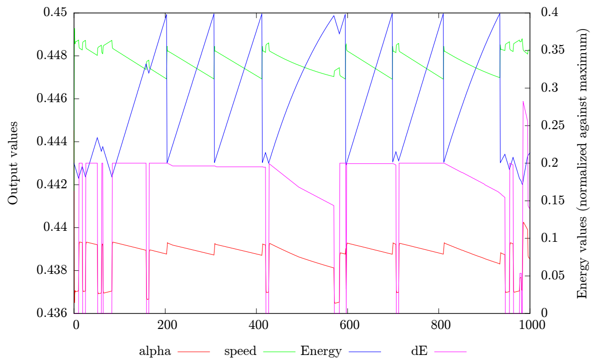


Figure 8: Typical outputs of Individual 17 over its lifetime. Those are interpreted to represent its angle and speed. The inputs leading to those decisions, the current energy and the current reward dE, are also represented. Note that the outputs varies very little over the agent lifetime, indicating that they might be extremely tuned to the present environment. Oscillations in energy show when the agent is circling around a food patch. When the reward of the patch starts to decrease, the agent moves to another patch.

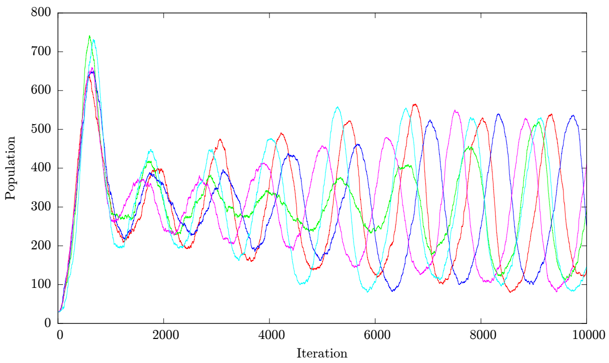


Figure 9: Total population in multiple runs where Individual 17 was the winner. Based on stochastic conditions at the beginning of the run, the oscillations are out of phase. Some runs may also take longer to reach the final amplitude.

A more conservative approach is to move away before the patch is completely drained, giving it time to regrow. The difference between behaviors then lies in the way energy is taken from the patch. The most aggressive of these is to simply gather on top of the patch, getting as much energy for oneself as possible. A more sustainable tactic is to take turns by circling near the patch, which can even allow the patch to recover if necessary. Finally, agents can try to minimize food depletion by taking only a small share of the energy available before moving to another area. As such, those can be seen as spatial and temporal implementation of mixed strategies in the Hawk-Dove game, giving them a physical interpretation.

In all but the most frugal populations, agents have a large impact on the amount of food available in the environment. This leads to oscillatory dynamics as the population grows until passes above the limit that can be sustained by the environment, then loses a number of individuals due to shortage of food, until the resource regrowth made it sufficient again. The amount of resource is therefore limiting the population as a carrying capacity, as described in White (1978).

By evolving agents separately the experiment really isolates populations, artificially evolving them in different ecological niches. Isolation has been hypothesized to help give rise to altruistic behavior (Cohen and Eshel 1976). Indeed, in an isolated population, the individuals have more chance to share common genes with one another, in turn amplifying their tendency to kin selection (Smith 1964), thus resulting in all individuals in a given isolated group adopting the cooperating, Dove-like behavior. When the population is then reintroduced in the initial population, the more efficient, cooperating behavior is susceptible to crystallize to the whole population from an inbred founder effect (Provine 2004, Sapolsky 2004).

Once we reinject different strategies in one environment, where they come into contact and compete with each other, we could observe two favorable approaches, leaning either toward frugal or greedy. While it is understandable that extreme strategies would perform more poorly than more adaptable ones, it is interesting to see that strategies that seem to be the most balanced are also inefficient. Another contribution in this paper is to show that, if we prevent agent populations from modifying their strategies through mutation, only collaborating (Dove-like) strategies remain stable among those two. The reasons can be many, ranging from an effective tuning of the strategy to the environment, making further mutation deleterious, to an ease to perform well in a variety of situations without further adjustment. On the other hand, since spore strategies tend to produce a large amount of offspring while food is available, those strategies may evolve faster, taking better advantage of mutations.

In future work, it would be interesting to investigate a much larger sampling of agents for the tournament. A preliminary test with 800 initial agents seems to show more diversity among victorious agents, with or without mutation of the genotype.

It might also be fruitful to investigate the impact of the density of food patches. As a negative control, agents going in straight line, but able to control their speed were able in 6 runs out of 10 to evolve a stable population, albeit much lower than that of normal agents. Decreasing the density to a level were lucky solutions are not available might change the distribution of evolved strategies, potentially favoring the more frugal ones. Additionally, generating an equivalent payoff matrix from those settings may give insights on the proportions of the various strategies, as well as the overwhelming winning rates of agents 16 and 17 in the tournament analysis.

Finally, agents are not sensing each other directly, and more complex behaviors are theoretically possible if a certain mode of signaling was introduced in the model. In terms of cooperation between agents, this would for example allow for mechanisms more complex, as agents may learn to recognize each other. Adding signal to our agents might yield richer cooperation dynamics among separate species with common tactics.

Acknowledgments

This paper was partially supported by Grant-in-Aid for Scientific Research on Innovative Areas (Research Project Number:15H01612). We would also like to thank Dr. Julien Hubert for helpful comments and discussions.

References

Aktipis, C. (2004). Know when to walk away: contingent movement and the evolution of cooperation. *Journal of Theoretical Biology*, 231(2):249–260.

Charnov, E. L. (1976). Optimal foraging, the marginal value theorem. *Theoretical population biology*, 9(2):129–136.

Cohen, D. and Eshel, I. (1976). On the founder effect and the evolution of altruistic traits. *Theoretical population biology*, 10(3):276–302.

Davies, N. B., Krebs, J. R., and West, S. A. (2012). *An introduction to behavioural ecology*. John Wiley & Sons.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

Fretwell, S. and Lucas, H. J. (1970). On territorial behavior and other factors influencing.

Glass, N. R. (1971). Computer analysis of predation energetics in the largemouth bass. *Systems analysis and simulation in ecology*, 1:325–363.

Gyllenberg, M. and Parvinen, K. (2001). Necessary and sufficient conditions for evolutionary suicide. *Bulletin of mathematical biology*, 63(5):981–993.

Hardin, G. (1968). The tragedy of the commons. *science*, 162(3859):1243–1248.

Huffaker, C. B., Herman, S., and Shea, K. (1963). *Experimental studies on predation: complex dispersion and levels of food in an acarine predator-prey interaction*. University of Calif.

Kerr, B., Riley, M. A., Feldman, M. W., and Bohannan, B. J. (2002). Local dispersal promotes biodiversity in a real-life game of rock–paper–scissors. *Nature*, 418(6894):171–174.

Kotler, B. P., Brown, J. S., Dall, S. R., Gresser, S., Ganey, D., and Bouskila, A. (2002). Foraging games between gerbils and their predators: temporal dynamics of resource depletion and apprehension in gerbils. *Evolutionary Ecology Research*, 4(4):495–518.

Lipsitch, M., Herre, E. A., and Nowak, M. A. (1995). Host population structure and the evolution of virulence: a" law of diminishing returns". *Evolution*, pages 743–748.

Lotka, A. J. (1910). Contribution to the theory of periodic reactions. *The Journal of Physical Chemistry*, 14(3):271–274.

MacArthur, R. H. and Pianka, E. R. (1966). On optimal use of a patchy environment. *American Naturalist*, pages 603–609.

Matsuda, H. and Abrams, P. A. (1994). Runaway evolution to self-extinction under asymmetrical competition. *Evolution*, pages 1764–1772.

Odling-Smee, F. J., Laland, K. N., and Feldman, M. W. (1996). Niche construction. *American Naturalist*, pages 641–648.

O'brien, W. J., Browman, H. I., and Evans, B. I. (1990). Search strategies of foraging animals. *American Scientist*, 78(2):152–160.

Provine, W. B. (2004). Ernst mayr genetics and speciation. *Genetics*, 167(3):1041–1046.

Pyke, G. H. (1984). Optimal foraging theory: a critical review. *Annual review of ecology and systematics*, pages 523–575.

Sapolsky, R. M. (2004). *Why zebras don't get ulcers: The acclaimed guide to stress, stress-related diseases, and coping-now revised and updated*. Macmillan.

Seth, A. K. (2007). The ecology of action selection: insights from artificial life. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1485):1545–1558.

Smith, J. M. (1964). Group selection and kin selection. *Nature*, 201:1145–1147.

Smith, J. M. (1974). The theory of games and the evolution of animal conflicts. *Journal of theoretical biology*, 47(1):209–221.

Smith, J. M. (1982). *Evolution and the Theory of Games*. Cambridge university press.

Stephens, D. W. and Krebs, J. R. (1986). Foraging theoryprinceton university press.

Sugden, R. (2004). *The economics of rights, co-operation and welfare*. Palgrave Macmillan Basingstoke.

Turchin, P. (2003). *Complex population dynamics: a theoretical/empirical synthesis*, volume 35. Princeton University Press.

Volterra, V. (1926). Fluctuations in the abundance of a species considered mathematically. *Nature*, 118:558–560.

Werner, E. E. and Hall, D. J. (1974). Optimal foraging and the size selection of prey by the bluegill sunfish (lepomis macrochirus). *Ecology*, pages 1042–1052.

White, T. (1978). The importance of a relative shortage of food in animal ecology. *Oecologia*, 33(1):71–86.

Witkowski, O. and Aubert, N. (2014). Pseudo-static cooperators: Moving isn't always about going somewhere. *Proceedings of the Fourteenth International Conference on the Simulation and Synthesis of Living Systems (Artificial Life 14)*, 14:392–397.

Witkowski, O. and Ikegami, T. (2014). Asynchronous evolution: Emergence of signal-based swarming. *Proceedings of the Fourteenth International Conference on the Simulation and Synthesis of Living Systems (Artificial Life 14)*, 14:302–309.

Wallace: An efficient generic evolutionary framework

Christopher Steven Timperley and Susan Stepney,
Department of Computer Science and York Centre for Complex Systems Analysis
University of York,
York, United Kingdom
ct584@york.ac.uk susan.stepney@york.ac.uk

Abstract

We present a novel evolutionary computing framework, Wallace, that achieves ease-of-use and genericity, via a domain-specific language, and simultaneously achieves efficiency via meta-programming, as well as supporting parallelism. Wallace also includes a novel multiple representation model of individual development, realised using meta-programming. We describe the Wallace framework, illustrating it with a number of example problems from the literature. We compare the performance of this framework to existing EC frameworks; early results show improvements in both conciseness and speed over popular alternatives. Finally, we discuss the future of EC frameworks, and the ongoing developments to the Wallace framework.

Introduction

Choosing the right evolutionary computation (EC) framework is a challenge; wherever high performance is offered, large amounts of "boilerplate" code in a low-level language seem sure to follow, whilst frameworks offering expressiveness through a simple and elegant syntax almost exclusively do so at the cost of performance, reducing their value to researchers and industrial users. By forcing users to decide between expressiveness and performance, these tools divide the research and development efforts of the community.

Our new EC framework, Wallace, achieves both ease-of-use and high performance. Wallace exploits the expressive power and conciseness of domain-specific languages (DSLs) with the process of computational reflection (Maes, 1987), the ability to write and modify parts of a program at run-time. Using the semantic information provided in human-readable descriptions, we synthesise highly optimised algorithms specific to the details of a given problem.

The rest of the paper is structured as follows. First, we review the current state of EC frameworks, examining the design decisions they present and the trade-offs they incur. We then propose how DSLs and meta-programming can be used to avoid these trade-offs, and introduce our own EC framework, Wallace. We discuss its architecture and key features, before providing a number of examples written in Wallace, then briefly examine the underlying meta-architecture used to implement it. Finally, we compare the brevity and performance of Wallace against that of some of the most popular EC frameworks.

Background

As the field of EC continues to grow in popularity, the number of tools and frameworks increases, each varying in many respects from the last, addressing the problems of a former generation, whilst creating a new set of problems for a future generation to address.

Examples of such problems include: sacrificing performance when implementing frameworks in a dynamic language; the search of brevity and ease-of-use; the introduction of overly complicated abstractions in an attempt to tailor to all known evolutionary algorithms, harming both performance and maintainability.

Where one tool appears to perform well in one respect, it often lacks in another; such compromises often determine the intended audience of the framework and ultimately constrain its applicability. Despite the constant introduction of new software in the EC community, some older frameworks, such as ECJ (Luke, nd) and Evolving Objects (Keijzer et al., 2002), have managed to maintain a lasting appeal. Why have these tools remained so popular, where have others failed, and what problems still remain to be solved?

Audience

There are three main audiences for EC tools: educational, industrial, and research. As frameworks increase in applicability, performance and genericity, they become more aligned to research and industrial users, who require the ability to write fast and highly custom algorithms for complex problems, but in the process they decrease their ease-of-use and raise their barrier to entry, making them less suited to an educational audience.

Applicability

Each framework varies in its domain of applicability; some are restricted to performing a small subset of EC, such as

GEVA (O'Neill et al., 2008), whilst others are built to facilitate all forms of EC. Each approach has its own merits and disadvantages.

By modelling only a subset of EC, one may reap the performance benefits of specificity, by significantly reducing the level of abstraction, removing inefficient generic code, and using more efficient memory allocation patterns. This improves both performance of the framework, and the maintainability of its codebase, but comes at the cost of learning a new framework for each form of EC one wishes to perform. Furthermore, creating a new framework for each field of EC involves re-implementing highly common operations, representations, and logging facilities.

More generic tools, such as ECJ and EO, are designed to be applicable in all realms of meta-heuristic computation, but at the cost of performance; their highly abstract frameworks add significant layers of overhead and complexity, removing much potential for optimisation. EO deals with this better than ECJ, by exploiting C++ templates, but increases the complexity of its language in the process, raising the barrier to entry. ECJ and EO possess plentiful libraries of operators, representations and more, but other high-level frameworks, such as JCLEC (Ventura et al., 2008) and Watchmaker (Dyer, 2010), are often lacking; it is easier to implement a small subset of EC well than it is to implement its entirety.

As highlighted by Gagné and Parizeau (2006), EC frameworks differ in their domains of applicability, and in the genericity of their various components, such as their representation, fitness, operations, evolutionary model, parameters, and output. Tools such as ECJ and EO have genericity in all criteria proposed by Gagné and Parizeau, and allow users to easily integrate new concepts into the framework.

Ease-of-Use

High performance frameworks are almost exclusively written in relatively low-level languages such as C, C++ and Java. Such frameworks naturally incur considerable boilerplate code and require an extensive knowledge of their underlying language. This higher level of performance comes at the cost of a reduced level of conciseness and expressiveness. By trading off these properties in search of performance, the barrier of entry to these tools is raised, and their potential as educational tools is diminished.

Frameworks written in interpreted languages, such as Python and Ruby, require fewer lines of code and permit more human readable descriptions than their compiled counterparts. Additionally, such languages allow the user to easily prototype, inspect and modify algorithms as they are running, making them well suited to the classroom. However, such advantages are attained by sacrificing the speed and optimisation opportunities afforded by faster compiled languages.

Some high performance frameworks escape the difficulties of dealing with low-level languages by bypassing them all together, and instead relying on user input to graphical user interfaces to setup and run algorithms. Whilst these GUIs often make for excellent educational tools, they are seldom useful to the industrial or research user, who almost always wishes to use representations, operations and algorithms beyond those incorporated within the tool.

Frameworks may attempt to escape these issues by allowing users to describe their algorithms in the form of a highly restricted DSL. Such descriptions serve as rigid specifications, say in the form of Java parameter files or XML files, allowing the user to specify the various settings of their algorithms from a set of predefined options. Whilst these descriptions are often more concise than their alternative, they are so at the cost of expressiveness; seldom is the user allowed to describe the behavioural aspects of an algorithm without writing in the language of the framework itself.

An Ideal Language

We believe that if a framework achieved ease-of-use, applicability, genericity *and* performance, that it might better serve all audiences, and thus become a candidate for a common language, bringing the EC audiences closer together. Yet such a framework seems impossible; surely one cannot maintain performance whilst enjoying genericity and applicability, which themselves cannot be enjoyed without compromise to ease-of-use?

Our solution is two-pronged. First, by employing a DSL, we can still enjoy ease-of-use, whilst maintain genericity and applicability. Such an approach not only allows the user to write algorithms in more natural terms, making them more amenable to communication, but also retains the expressiveness of the underlying programming language, allowing the user to integrate behaviours beyond those prescribed, unlike parameter file based approaches.

Second, in order to maintain high performance whilst achieving these qualities, we exploit meta-programming to allow us to write high level code that remains on a par with low level approaches in terms of performance. Rather than using our DSL to simply specify components within the language, we enhance it with the ability to extend the language and to dynamically synthesise new code, allowing natural high-level descriptions to be transformed into highly optimised context-aware code fragments on-the-fly, as shown in Figure 1.

Our realisation of this solution is the **Wallace** framework, named after the famous naturalist and co-discoverer of natural selection, Alfred Russel Wallace. To achieve these feats we implemented Wallace using the Julia language.

Julia (Bezanson et al., 2014) is a relatively young high-level, high-performance dynamic programming language, designed for technical computing, built around multiple-dispatch, a rich type system, and a just-in-time compiler that specialises methods based upon types encountered at run-

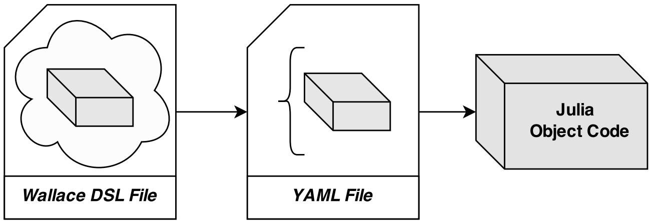


Figure 1: High-level descriptions of objects are parsed into YAML before being dynamically synthesized into optimised Julia object code.

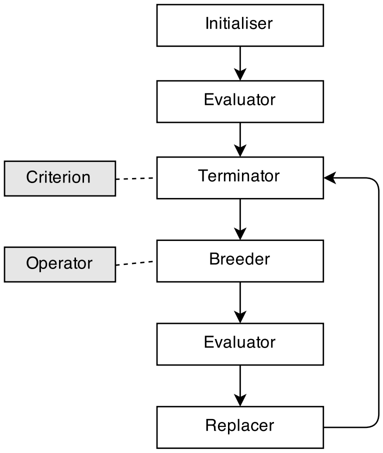


Figure 2: The flow of data within the component-based architecture of an EA in Wallace.

time. As a result of these design decisions, Julia attains a performance close to if not equal to C across a wide range of problems, without needing to resort to writing low-level code (Bezanson et al., 2012).

Architecture

Similar to ECJ and EO, Wallace employs a component-based architecture, where algorithms are described in terms of a series of customisable components, each responsible for implementing some part of the standard evolutionary loop employed by the majority of EAs; the flow of data between the components within this architecture is shown in Figure 2. Within Wallace each component is built using a provided description, and often compiled to a highly specific and optimised form through the use of run-time code evaluation.

In this section we outline how Wallace implements some of these components, and discuss its population model and its novel multiple representation model.

Population Model

Wallace uses a population model similar to that of EO and ECJ, where an algorithm operates on a single population divided into an arbitrary number of independently evolving sub-populations, or *demes*.

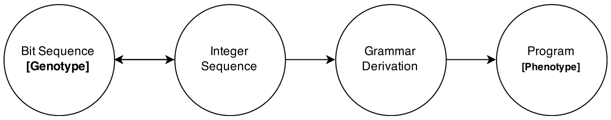


Figure 3: An example series of developmental stages for a problem using grammatical evolution. A sequence of bits is used as the genotype of an individual, from which an integer sequence may be produced; the bi-directional indicates that changes in the integer sequence may be transmitted to the bit sequence. From the integer sequence, a grammar derivation is produced, which in turn, is used to produce a program.

```
species:
  stages:
    bit_sequence:
      representation<bit_vector>: { length: 800 }
    int_sequence:
      from: bit_sequence
      representation<int_vector>: { length: 100 }
    derivation:
      from: int_sequence
      representation<string>: { frozen: true }
    program:
      from: derivation
      representation<lambda>:
        arguments: ["x::Int", "y", "z"]
```

Figure 4: A multiple representation realisation of a simple grammatical evolution (Figure 3) approach to symbolic regression.

Each deme hosts a single *species* of individuals. The population may comprise several heterogeneous demes, allowing different species to be co-evolved. Island model EAs can be realised by attaching a *migrator* component to the population, which exchanges individuals between demes, according to a *migration policy* (Whitley et al., 1998). By combining the migrator component with Wallace's multiple representation model (below), we can implement more advanced models, such as the multiple representation island model (Skolicki and De Jong, 2004), where each deme evolves a different representation of a given problem in parallel.

Multiple Representation Model

A novel feature of Wallace is its multiple representation model, which allows individuals to include an arbitrary number of linked representations, implementing a rich process of *development*. One may create a *Lamarckian* connection between certain developmental stages, allowing changes made to a later stage to be communicated back to earlier ones. An example is given in Figure 3.

To define a species, the user details the development stages of its individuals, by specifying the representation used by each stage, along with any associated parameters, and the development stage from which it should be produced. An example species definition is given in Figure 4.

Users may add arbitrary representations into Wallace by registering their implementing type and associated factory with the kernel. These representation types define the de-

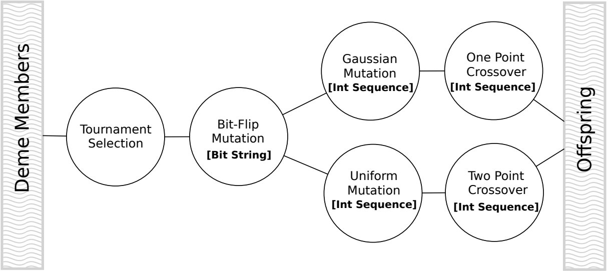

Deme Members
Tournament Selection
Bit-Flip Mutation [Bit String]
Gaussian Mutation [Int Sequence]
One Point Crossover [Int Sequence]
Uniform Mutation [Int Sequence]
Two Point Crossover [Int Sequence]
Offspring

Figure 5: An example breeding setup, illustrating the chain of operations that a proto-offspring is subjected to before being inserted into the set of offspring. Each node within the graph represents a particular selection method, or a variation method, in which case it operates on the developmental stage described within.

fault way to pseudo-randomly generate new individuals, and are used to describe how an instance of one representation should be transformed into an instance of another.

Breeding Model

Selecting parents and producing offspring is the responsibility of Wallace's *breeder* component, heavily inspired by ECJ's powerful breeding pipeline system. Each deme is allocated its own breeder, allowing different demes to carry out their own process of breeding. This breeder produces a set of offspring at each generation, prior to the stage of *replacement*, where the *replacement* component decides which individuals from the current members of the deme and its set of offspring should survive into the next generation.

The simplest breeder is the *fast breeder*, which takes a single breeding source and uses it to produce a desired number of offspring. This breeding source may take the form of a selection or variation method, or as a form of proxy, either returning offspring from a number of different sources based upon chance, or selecting between them based upon the state of the search. Each source may draw its inputs from other sources, or directly from the contents of a deme, as in the case of selection methods. An example breeding setup is shown in Figure 5.

Using the syntax in Figure 6, users may add new sources into the breeder by declaring their type (e.g. selector or variation operator), the name of the source from which they should draw their inputs, the name of the developmental stage upon which they should operate, which defaults to the genotype if omitted, along with any operator-specific parameters, such as mutation rate and tournament size.

Like most breeders within Wallace, the fast breeder has support for parallel breeding, which it achieves by splitting the workload at each step within the breeding process as equally as possible between all available cores.

Fitness Model

Following breeding, and prior to replacement, each unevaluated candidate solution within the population is subject to

```
breeder<breeder/fast>:
  sources:
    s<selection>:
      operator<selection/tournament>: { size: 2 }
    x<variation>:
      from: s
      stage: bits
      operator<crossover/two_point>: { rate: 0.7 }
  m<variation>:
    from: x
    stage: bits
    operator<mutation/bit_flip> { rate: 0.01 }
```

Figure 6: An example of a fast breeder setup.

```
evaluator<evaluator/simple>:
  objective: |
    SimpleFitness(true, count(i.bit_vector, 1))
```

Figure 7: An example of a simple evaluator, used to compute fitness as the number of ones within an individual's bit vector representation.

evaluation. This process is carried out by an *evaluator*, which accepts a population and assigns fitness values to all of its (unevaluated) individuals based on their performance. This evaluator may treat each individual in isolation, or may implement a co-evolutionary approach, where the fitness of an individual is calculated by combining or comparing it against other individuals in the same population, or another population.

The *simple* evaluator is built by being given an objective function, provided in the form of a lambda function, accepting a single input individual and returning a calculated Fitness object for that individual (Figure 7).

The concept of fitness itself is implemented using abstract Fitness objects, which only require the programmer to specify a means of comparing them, allowing multiple-objective and more advanced co-evolutionary fitness values to be used; this avoids the user becoming trapped within the limitations of a single scalar fitness value, as is the case with many other frameworks.

In a similar fashion to breeders, evaluators may also take advantage of multiple cores by splitting their workload across each of them.

Examples

To illustrate the use of Wallace, we give two examples: implementing the OneMax problem using a simple genetic algorithm (Figure 8), and evolving a Java AI controller for the Robocode (Nelson et al., 2014) tank game via grammatical evolution (Figure 9).

We begin the algorithm description for a problem by starting with the algorithm keyword, followed by the name of our particular algorithm. We specify that our algorithm extends the base evolutionary algorithm; were we to imple-

```
type: evolutionary_algorithm

_my_breeder<breeder/fast>:
  sources:
    s<selection>:
      operator<selection/tournament> { size: 2 }
    x<variation>:
      from: s
      stage: bits
      operator<crossover/uniform>: { rate: 0.7 }
    m<variation>:
      from: x
      stage: bits
      operator<mutation/bit_flip>: { rate: 0.01 }

_my_species:
  stages:
    bits:
      representation<bit_vector>: { length: 100 }

replacement<generational>: {}

population:
  demes:
    - capacity: 100
      species:  $(_my_species)
      breeder:  $(_my_breeder)

evaluator<evaluator/simple>:
  objective: |
    SimpleFitness(sum(i.bits), true)
```

Figure 8: An implementation of OneMax within Wallace.

ment some other form of meta-heuristic we would extend its base type instead.

We then describe the species of individuals that we wish to evolve, outlining each of its stages of development and the representations they use, as well as the type of fitness object used to compare the fitness of individuals within the same species. For OneMax, the species definition says that our individuals have one stage of development, that being their bit-string, and that we use a simple scalar-based measure of fitness. For the more complex Robocode problem, we setup the bit-string, codon sequence, and grammar derivation, representing the source code for our robot controller, as commonly used when performing GE, along with an additional controller stage that we have added, which uses a custom representation to simplify compilation of robocode controllers and communications with the robocode platform.

Next, we outline the specifics of our breeder, detailing each of its operators, along with their respective sources and the name of the development stage on which they operate, in the case of variation methods.

Finally, we specify our method of replacing individuals following the breeding phase, and specify the size, species and breeder used by each of the demes within the population, before providing an evaluation function for our problem, responsible for measuring the quality of potential solutions.

Wallace takes these descriptions and using meta-programming, compiles them into highly efficient object-code, optimised to the specifics of the given problem.

```
type: evolutionary_algorithm

termination:
  iterations<criterion/iterations>: { limit: 1000 }

_my_species:
  stages:
    bits:
      representation<bit_vector>: { length: 800 }
    codons:
      from: bits
      lamarckian: true
      representation<int_vector>: { length: 100 }
    source_code:
      from: codons
      representation<grammar_derivation>:
        root:  s
        rules: ...
    controller:
      from: source_code
      representation<robocode_controller>: {}

_my_breeder<breeder/fast>:
  sources:
    s<selection>:
      operator<selection/tournament>: { size: 5 }
    x<variation>:
      from:   s
      stage:  bits
      operator<crossover/two_point>: { rate: 0.7 }
    m<variation>:
      from: x
      stage: codons
      operator<mutation/gaussian> { rate: 0.02 }

replacement<generational>: { elites: 1 }

population:
  demes:
    - capacity: 100
      species: $(_my_species)
      breeder: $(_my_breeder)

evaluator<evaluator/simple>:
  objective: |
    score = execute(i.controller)
    SimpleFitness(score, true)
```

Figure 9: An implementation of Robocode controller evolution within Wallace.

Meta-Architecture

The Wallace meta-architecture provides a process responsible for transforming concise high-level component descriptions, provided in a DSL, into fragments of highly optimised code, using meta-programming. It is through the reflective capabilities of Julia that Wallace manages to perform such transformations on-the-fly, as the program is running.

Wallace Description Language

Rather than interacting directly with the Julia programming language, Wallace users provide natural and concise high-level descriptions of their algorithms and their various components using a specialised DSL built upon an extended YAML, which retains the ability to provide code fragments and to define new behaviours.

Descriptions are realised as objects within Wallace by passing them to the make function. Following a pattern sim-

```
type: evolutionary_algorithm

evaluator:
  type: evaluator/simple
  objective: |
    ...

replacement:
  type: generational
...
```

Figure 10: An example algorithm description translated to YAML.

```
> julia
Julia...

julia> using Wallace

wallace> alg = load("max_ones.wlc")
<EvolutionaryAlgorithm:...>

wallace> run!(alg)
...
```

Figure 11: Loading and executing an algorithm description via the REPL.

ilar to that of the Factory pattern (Freeman et al., 2004), inspired by the popular Ruby test data generation tool, FactoryGirl (Ferris, 2014), descriptions are forwarded to the factory responsible for unfolding them; this may be done automatically by the Wallace kernel, by inspecting the type attribute of the description, or by manually specifying which factory should be used.

Once a description has reached a factory, it is subject to the processes of *preparation*, *validation* and *composition*, before it is transformed into an object of the desired type (which may itself be a description).

Preparation Stage The description is transformed into a valid YAML object (Figure 10), by first removing all comments, before extracting each type tag from within the document and inserting it into its associated object, and finally handling all insertion point operations.

Validation Stage Next, the YAML description is passed to the validation stage, where it is checked against a series of rules for legality, provided in the form of a function, before it passed onto the final stage to be transformed into a concrete object.

Composition Stage Finally, the validated description is used to construct the concrete object it describes. Usually this stage involves recursively constructing the concrete objects of the desired object's individual components, by invoking each of their associated factories with their descriptions, before composing them into a single object. Rather than provide a concrete object, it may be the case that certain *abstract* factories are used to construct partial objects, or to further detail given descriptions.

Performance Optimisations

In addition to using reflection to transform descriptions into object code, Wallace exploits reflection to implement optimised data structures and algorithms, highly specific to the details of a given problem.

To realise its multiple representation model without incurring a performance hit by storing each stage of an individual's development in an abstract container, Wallace uses the information provided by a description of a given species to create a memory-optimised individual type, specific to that species; this approach substantially reduces look-up time and memory consumption. This technique is also applied to the fitness model used by a given species of individuals; by specifying the type of fitness used by individuals within that species, Wallace can create a faster and more compact individual type definition by stating it specifically.

Wallace also uses reflection to implement a more efficient mechanism for converting between representations. Rather than dynamically calculating whether a given stage is out of sync with the genome and then determining the series of conversion steps that should be performed, Wallace determines the state of each of the development stages following each operation and hard-codes the minimal number of conversion operations into a highly specific (and thus optimised) breeder.

Typically, these conversion operations are applied to each individual within a group in sequence, and written in terms of a mapping operation on a single individual. However, the user may elect to implement a *batch conversion* mechanism for their representation, defining how a group of individuals should have representations converted collectively, allowing costly overheads to be minimised, such as compiling external code, proving useful when performing grammatical evolution in an external language.

Framework Interaction

Currently, Wallace lets its users interact in three different ways, allowing it to be used in a variety of contexts and for a multitude of purposes.

Read-Eval-Print-Loop (REPL) The quickest way to get up and running with Wallace is through Julia's REPL interface, augmented with additional functionality, allowing users to quickly perform experiments and prototype simple code. Component descriptions can be loaded into Wallace through calling the load or build commands with the location of a valid component description file (Figure 11). Once loaded, objects may be treated as any other kind of object within Julia, allowing them to interact with external libraries.

The REPL also provides a number of usability-driven fea-

Framework	Language
ECJ	Java
JCLEC	Java
DEAP	Python
Pyevolve	Python
inspyred	Python

Table 1: Evolutionary computation frameworks to perform performance and brevity comparisons.

tures, some of which are listed below, which allow users to quickly assimilate themselves with the framework with minimal effort.

- help(type), used to provide descriptions of the various components available within Wallace, along with their inputs and modes of operations, in a neat and user-friendly manner.
- list_subtypes(type), provides a list of all known subtypes of a given type, each of which can be further inspected using the help command.

Script Execution One may also write conventional Julia scripts to compose and execute their algorithms, by simply calling "using Wallace" at the top of the script, and executing the file as standard from the command line.

Interactive Graphical Notebook As Wallace seamlessly integrates into the Julia language, one may also exploit Julia's powerful browser-based interactive environment, IJulia, built upon the popular IPython/Jupyter interactive computational environment. By combining IJulia with other packages, such as Gadfly and Interact.jl, one may also create a large variety of highly detailed and customisable plots and graphs.

Together, these tools form a powerful package for evolutionary computation, both in a research, industrial and classroom environment, allowing experienced users and newcomers alike to quickly and empirically prototype solutions and to visualise the evolutionary process, whilst retaining the performance benefits shared by lower level languages.

Comparison

Here we compare the brevity and performance of Wallace to a selection of the most popular EC frameworks used within the literature, listed in Table 1, in order to assess its standing. Due to time constraints and compilation issues, we were unfortunately unable to test performance against other C/C++ options.

Brevity

In order to compare the conciseness of Wallace descriptions to those of other frameworks, we repeated the study carried out by Fortin et al. (2012), where the number of lines required to implement an algorithm to solve the OneMax benchmark problem is counted and compared for each framework. The results are given in Table 2.

Framework	Type	Config.	Alg.	Example	Total
Wallace	34	29	22	0	85
ECJ	308	34	88	26	456
Pyevolve	59	0	261	16	336
inspyred	0	0	330	30	360
DEAP	0	0	0	59	59
JCLEC	198	15	192	29	434

Table 2: Number of lines required to implement OneMax problem using different EC frameworks.

	Benchmark
GA1	OneMax ($n = 100$)
GA2	Rastrigin ($n = 100$)
GP1	Artificial Ant (Santa-Fe Trail, 400 moves)
GP2	Symbolic Regression ($x^4 + x^3 + x^2 + x$)
GP3	Boolean Circuit (8x3 multiplexer)

Table 3: Benchmark problems used to compare performance of different EC frameworks. Each had a population size of 100, and ran for 1000 generations.

Whilst it took more lines in Wallace to implement the OneMax problem than it did using DEAP, a framework firmly established for its clarity and brevity, the example file used to enter the specifics of the algorithm setup and the OneMax problem was smaller and considerably simpler than that used by DEAP, involving only the specification of problem details in a simple and compact structure.

Furthermore, Wallace required only that the user describe the setup of their algorithm in terms of its components, allowing them to choose from a range of operators and representations from its standard library, whereas the example file used by DEAP required the user to write the EA from scratch in Python, using its framework to skip boilerplate definitions.

Performance

In order to fairly assess the performance of Wallace to that of other frameworks, we compared execution times across 100 runs for a number of common benchmark functions on a single thread, listed in Table 3, under the same conditions and using the same algorithm setup (or as close as the underlying framework would permit).

An overview of the results from our experiment are given in Table 4; the full experiment setup and raw results data are online at https://github.com/ChrisTimperley/EC-Software-Benchmarks. Where it was not possible to implement a solution to a given problem using the standard library of a framework, that benchmark was skipped, and is denoted within the table by *n/a*.

Framework	GA1	GA2	GP1	GP2	GP3
Wallace	0.329	0.453	0.938	0.294	15.971
ECJ	0.502	1.070	1.338	0.791	31.759
JCLEC	0.386	0.431	n/a	0.430	n/a
DEAP	5.357	12.530	105.683	53.949	59.056
Inspyred	5.343	11.679	n/a	n/a	n/a
Pyevolve	2.977	5.011	n/a	n/a	n/a

Table 4: Mean time taken to perform each benchmark, averaged over 100 runs. Recorded using the @time macro in Julia for Wallace, and using time in the shell for all others.

As one might predict, the results show a marked difference between the relatively slow performance of the Python-based frameworks and the significantly faster Java-based frameworks.

However, Wallace, like the Python-based frameworks, is written in a high-level dynamic language, yet it shows a performance similar to, and in some cases beyond that of, the the lower-level Java-based frameworks. Wallace attains the best performance on 4 out of the 5 benchmarks tried, with only a narrow gap to the winner of GA2, JCLEC. Although difference in performance between Wallace and JCLEC is relatively small across each of the benchmarks, the difference between Wallace and each of the other languages tested is far more noticeable.

Conclusion

We have presented Wallace, a new EC framework, possessing both high performance *and* a high degree of conciseness and clarity, through the novel combination of DSLs and computational reflection. The results of our comparison show that Wallace shares a similar performance with the fastest existing frameworks that we analysed, and manages to do so whilst requiring substantially fewer lines of code than its nearest competitors.

The source code for Wallace, available under the LGPL license, along with its latest binaries, documentation and bug reporting can be found online at: https://github.com/ChrisTimperley/Wallace.jl.

Future Work

Our primary focus for the future remains on improving and supporting the Wallace framework so that users can easily write and share their own algorithms, representations and extensions. In the short term, we intend to do this by creating a dedicated website, complete with in-depth documentation and a series of tutorials for a wider range of users. In the further future, we hope to build upon Julia's package system and provide our own online repository, allowing users to share their creations for the benefit of the EC community.

We also intend to explore the wider possibilities for meta-programming within Wallace, and how our techniques might be applied to provide similar performance boosts to other large frameworks.

Acknowledgments

We thank Dan Franks, for his advice and support, and for trusting us to use Wallace for his Masters-level evolutionary computation module, and the patient students who provided us with invaluable feedback. We also thank Paul Andrews and Simon Hickinbotham, for numerous discussions about the ideas and direction of the framework. This work was partly funded by an EPSRC DTG.

References

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B. (2014). Julia: A fresh approach to numerical computing. *CoRR*, abs/1411.1607.

Bezanson, J., Karpinski, S., Shah, V. B., and Edelman, A. (2012). Julia: A fast dynamic language for technical computing. *CoRR*, abs/1209.5145.

Dyer, D. (2010). The Watchmaker Framework for Evolutionary Computation. http://watchmaker.uncommons.org/. Accessed: 2015-03-05.

Ferris, J. (2014). FactoryGirl. https://github.com/thoughtbot/factory_girl. Accessed: 2015-03-05.

Fortin, F.-A., De Rainville, F.-M., Gardner, M.-A., Parizeau, M., and Gagné, C. (2012). DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175.

Freeman, E., Freeman, E., Bates, B., and Sierra, K. (2004). *Head First Design Patterns*. O' Reilly & Associates, Inc.

Gagné, C. and Parizeau, M. (2006). Genericity in evolutionary computation software tools: Principles and case-study. *Int. J. on Artificial Intelligence Tools*, 15(02):173–194.

Keijzer, M., Merelo, J., Romero, G., and Schoenauer, M. (2002). Evolving Objects: A General Purpose Evolutionary Computation Library. In *Artificial Evolution*, volume 2310 of *LNCS*, pages 231–242. Springer.

Luke, S. (n.d.). ECJ 22: A Java-based Evolutionary Computation Research System. http://cs.gmu.edu/~eclab/projects/ecj/. Accessed: 2015-03-05.

Maes, P. (1987). Concepts and Experiments in Computational Reflection. In *OOPSLA '87*, pages 147–155. ACM.

Nelson, M. A., Larsen, F. N., and Savara, P. (2014). Robocode. http://robocode.sourceforge.net/. Accessed: 2015-03-05.

O'Neill, M., Hemberg, E., Gilligan, C., Bartley, E., McDermott, J., and Brabazon, A. (2008). GEVA: Grammatical Evolution in Java. *SIGEVOLution*, 3(2):17–22.

Skolicki, Z. and De Jong, K. (2004). Improving Evolutionary Algorithms with Multi-representation Island Models. In *PPSN VIII*, volume 3242 of *LNCS*, pages 420–429. Springer.

Ventura, S., Romero, C., Zafra, A., Delgado, J. A., and Hervás, C. (2008). JCLEC: a Java framework for evolutionary computation. *Soft Computing*, 12(4):381–392.

Whitley, D., Rana, S., and Heckendorn, R. B. (1998). The Island Model Genetic Algorithm: On Separability, Population Size and Convergence. *J. Comp. Info. Tech.*, 7:33–47.

Emergence of Sense-Making Behavior by the Stimulus Avoidance Principle: Experiments on a Robot Behavior Controlled by Cultured Neuronal Cells.

Atsushi Masumori[1], Norihiro Maruyama[1] Lana Sinapayen[1], Takeshi Mita[1],
Urs Frey[2], Douglas Bakkum[3], Hirokazu Takahashi[1] and Takashi Ikegami[1]

[1]The University of Tokyo, [2]RIKEN QBiC, [3]ETH Zürich
masumori@sacral.c.u-tokyo.ac.jp

Abstract

Robot experiments using real cultured neuronal cells as controllers are a way to explore the idea of embodied cognition. Real cultured neuronal cells have innate plasticity, and a sensorimotor coupling is expected to develop a neural circuit. Previous studies have suggested that a dissociated neuronal culture has two properties: i) *modifiability* of connection between neurons by external stimuli and ii) *stability* of the connection without external stimuli. If cultured neuronal cells are embodied by coupling to an environment, they learn to avoid external stimulation. We call this mechanism a "learning by stimulation avoidance" principle. We try to demonstrate that adaptive behavior, like wall avoidance, can emerge spontaneously from embodied cultured neuronal cells. In this study, we developed a system in which a robot moves in a real environment and is controlled by cultured neuronal cells growing on a glass plate. We used a high-density complementary metal-oxide-semiconductor array to monitor the neural dynamics. We then conducted a robotic experiment using this platform. The results showed that wall-avoidance behavior by a robot can be enhanced spontaneously without giving any reward from the external environment.

Introduction

Learning is a remarkable phenomenon in the neural system, and it is crucial for animals as embodied neural systems to learn adaptive behavior autonomously to survive. A key concept in studying adaptive behavior is homeostasis. Ashby argued that an adaptive behavior is just an outcome of a homeostatic property of a living system (Ashby (1960)). Di Paolo and Iizuka reported that adaptive behavior is an indispensable outcome of homeostatic neural dynamics (Di Paolo (2000); Iizuka and Di Paolo (2007); Di Paolo and Iizuka (2008)). Yet, those models are still too abstract to be tested in realistic situations.

Biological neural networks cultured in vitro can be used to study potential memory and learning by nervous systems. Using the real biological neural networks is advantageous in that, for example, we can study potential complexity, which may be difficult to implement in artificial neural networks. In this study, we use a dissociated cultured neural system as a model of a biological neural system. Although such cultured neural systems are much simpler than real brain systems, they have some important essential properties, including spontaneous activity, plasticity and rich and complex controllability. Homeostatic control may be one such property.

It has become easier and more popular to study the coupling between cultured neuronal cells and external systems (DeMarse and Dockendorf (2005), Novellino et al. (2007); Pizzi et al. (2009); Warwick (2010)). In previous studies, cultured neurons were connected to an external system, such as a mobile robot in a real space. The sensory information coming from the external system was used to stimulate the neuronal cells, and the resulting neural activities controlled the external system. This change of external system provided feedback to the neural cell states, and this process could be repeated. We call this a "closed loop" and regard it as a model of primitive sensorimotor couplings. By studying such closed-loop systems, we examine a biological memory, learning, or adaptability of a neural system with respect to embodiment.

Studies of closed-loop systems have been documented. For example, Bakkum et al. (2008) trained cultured neuronal cells to achieve a desired behavior with multiple stimulations. Hayashi et al. (2011) proposed another method that used a cultured neural system to incrementally learn to respond in a particular way to a particular input. One drawback of these studies is that they used a conventional microelectrode array as a recording device; this type of device does not have sufficient spatial resolution so that it is difficult to stimulate and accurately detect a single neuronal state. In order to overcome the drawbacks, we use a recently developed device (high-density microelectrode array using complementary metal-oxide semiconductor [CMOS] technology) to detect activities of individual neurons with high precision. The details of this recording device are described in the following sections. The other drawback in closed-loop studies is that an external evaluation function must be prepared and designed properly. A unique feature of our study is the examination of the self-development of such an evaluation function from the closed-loop system itself. We included this feature because we believe that self-development of an evaluation function is how adaptive behavior emerges spontaneously with most animals.

A principle of our experimental study stems from Shahaf and Marom (2001)'s pioneering work on a cultured neural system. They argued that cultured neuronal cells have the following two characteristics: i) *modifiability* of connection between neurons by external stimuli and ii) *stability* of the connection without external stimuli. When a system is coupled to a body, these properties of neural cells lead to intelligent behavior. We rephrase the above properties in the following way:

1 Providing external stimuli to cultured neuronal cells.

2 Connection between each neuron is changed by the external stimuli, and thus a behavior represented through the external body is also changed (*modifiability*).

3 If a behavior that can stop the external stimuli occurs, stimulation is finished and the current connection is stabilized (*Stability*).

4 By repeating the above processes, behavior that can avoid the stimulation is improved.

In this way, behavior to avoid a stimulation can emerge spontaneously without having any explicit reward or evaluation function. We call this a "learning by stimulation avoidance" (LSA) principle. LSA assures a homeostatic property as it sustains stability and variation simultaneously. Shahaf and Marom (2001) demonstrated that cultured neuronal cells can actually learn a desired activity pattern in a minimal closed-loop experiment where electrical stimulation is applied to neuronal cells and the stimulation is removed when the network shows a desired activity pattern. In this example, although the experimenters gave a desired neural activity pattern to explicitly remove stimulation, stimulation avoidance led to adaptive behavior. In the present study, we demonstrate that cultured neuronal cells, by coupling with a mobile robot, can learn an adaptive behavior through the LSA principle, without any explicit reward.

Materials and Methods

Dissociated neuronal culture

The neural cultures were prepared from the cerebral cortex of E18 Wistar rats. The cortex region was trypsinized with 0.25% trypsin, and the dissociated cells were plated and cultured on a recording device. The surface of the electrodes on the device was coated with 0.05% polyethylenimine and laminin for improving plating efficiency. The cells were cultured in Neurobasal Medium (Life Technologies) containing 10% L-Glutamine (Life Technologies) and 2% B27 supplement (Life Technologies) for the first 24 h. Half of the plating medium was replaced with growth medium (Dulbeccos modified Eagles medium (Life Technologies) containing 10% horse serum, 0.5 mM GlutaMAX (Life Technologies), and 1 mM sodium pyruvate) after the first 24 h. The cultures were placed in an incubator at 37 °C with an H_2O-saturated atmosphere consisting of 95% air and 5% CO_2. During cell culturing, half of the medium was replaced once after several days with the growth medium.

High-density micro electrode array

A high-density CMOS electrode array (Frey et al. (2010)) was used for measuring the extracellular electrophysiological activity of the cultured neurons (Figure 1). This CMOS array is superior to the conventional multielectrode array (MEA) used previously (Potter and DeMarse (2001); Eytan and Marom (2006); Madhavan et al. (2007)) in that it has far higher spatio-temporal resolution. The number of electrodes in conventional MEAs is small, usually about 64, and the locations of the recording electrodes are predetermined with an inter-electrode distance of about 200 m; thus, it is difficult to identify signals from an individual cell. In contrast, the CMOS arrays have 11,011 electrodes. The diameter of the electrode is 7 μm with an inter-electrode distance of 18 μm over an area of 1.8 mm 1.8 mm. It can record electrical activity on 126 electrodes at one time at a sampling rate of 20 kHz.

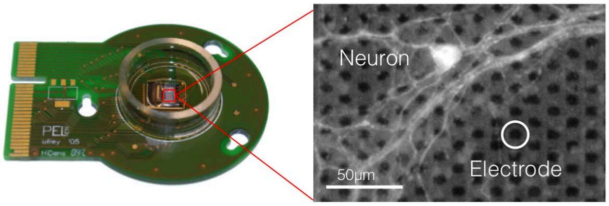


Figure 1: The high-density CMOS electrode array used in this experiment. This recording device has 11,011 recording sites, a diameter of 7 μm, and an inter-electrode distance of 18 μm.

Processing before and in recording neural activity

Before recording the neural activities, we scanned almost all the 11,011 electrodes on the CMOS array to obtain an electrical activity map for estimating the locations of the neuronal somata (i.e., identifying the positions of neural cells). In each of the 95 recording sessions, the electrical activities were recorded for 60 s with about 110 electrodes at the same time. An electrical activity map was obtained by averaging the height of the action potentials for each electrode. We applied a Gaussian filter to the map and assumed that the neuronal somata were located near the local peaks in the Gaussian-filtered map. About 120 of the higher level peaks were selected as the positions of neural cells, and the nearest electrodes to the peaks were selected for recording that neural activity. If the number of local peaks were fewer than 126, then all the peaks were used.

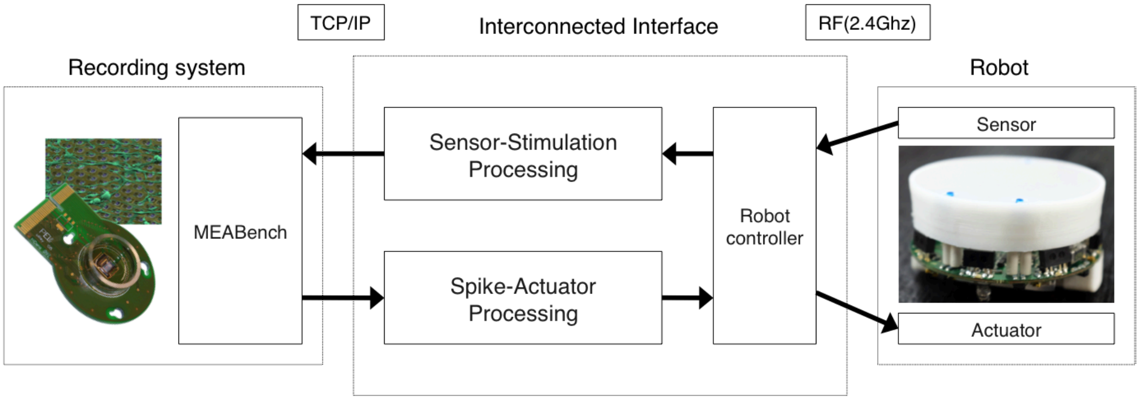


Figure 2: Overview of the closed-loop system composed of the high-density CMOS electrode array monitoring the culture of neuronal cells, a mobile robot, and the interface connecting them.

By using the above method, one electrode can ideally represent a single neural state. A type of neuronal cell is classified as excitatory or inhibitory, which is estimated by using the spike time series recorded for 10 min before the main experiment. Because the shapes of the action potential of these two neural types differ, we classified the type of neuronal cell by using k-means clustering.

For detecting and recording the spike of cultured neurons, we used the MEABench software developed by (Wagenaar et al. (2005)). All recordings were performed at a 20-kHz sampling rate using the real-time spike detection algorithm LimAda in MEABench. As this LimAda algorithm detects a spike that exceeds the threshold without distinction of positive and negative value, unexpected double detection of spikes can occur. These unexpected double-detected spikes were removed from the data before analyzing the data. By sending the electrical stimuli to a neuronal cell through the electrodes, such artifacts might occur. In a robotic experiment, we need to detect the action potential and stimulate the cultured neuronal cell at the same time. The Salpa filter in MEABench was used to remove the artifact in real time (Wagenaar and Potter (2002)).

A colsed-loop system

We implemented a closed-loop system between the cultured neural cells and a robot. This system mainly consisted of three components: a recording system monitoring the cultured neurons, a mobile robot as an external body of the cultured neurons, and the interface connecting them. A current system setup is depicted in Figure 2.

We used the CMOS array and the MEABench software for recording and stimulating neural cells. Elisa-3 (GCtronic, Ticino, Switzerland) was used as a mobile robot. Elisa-3 is a circular small robot of 2.5 cm radius and has two independently controllable wheels. The front right- and left-distance sensors were used as sensory signals for stimulating the neuronal cells. The refresh rate of the robot was set at 10 fps. The interface plays a role in receiving a sensor value from the robot and stimulating the neuronal cells based on the sensor value thorough the CMOS array. The interface also plays a role in receiving detected spike data from the CMOS array in real time and calculating a wheel speed based on the spike data and sending it to the robot. In this way, the robot and the neuronal cells form a closed loop. More details of the sensorimotor mapping are described in the following section.

Sensorimotor mapping

A simple sensorimotor mapping was applied to the robot and neuronal cells on the CMOS array (Figure 3). We selected two electrodes that were estimated as excitatory neurons as the left-and right-input neurons for sending the electrical stimuli. At given time intervals (100 ms), the probability $P_{L,R}$ for sending an electrical stimulation to the input neuron was controlled by the sensory value of the mobile robot. More practically, the probability is calculated as follows:

$$P_{L,R} = \begin{cases} 0 & (S_{L,R} < T) \\ S_{L,R}/S_{max} & (S_{L,R} \geq T) \end{cases}$$

If sensor value $S_{L,R}$ is less than a threshold T, $P_{L,R}$ becomes zero. Otherwise $P_{L,R}$ is calculated by $S_{L,R}/S_{max}$. S_{max} represents a maximum value of the sensor input. Whether an electrical stimulation is sent to the input neuron or not is determined with this probability every 100 ms.

We also selected 20 electrodes, 10 of which were left-output neurons and the other 10 were right ones; all 20 were within the vicinity of each input neuron for calculating each left- and right-wheel speed. The wheel speeds were calculated based on the number of spikes of the output neurons that were integrated every 100 ms. We calculated the left- and right-wheel speeds $V_{l,r}$ as follows:

$$V_{l,r} = \sum_{i \in \mathbf{N_{l,r}}} \omega_i v_i + C$$

These virtual neural states v_i take positive integers, which are equal to the number of spikes of the output neurons over a given time interval, and sum them with the fixed weight ω_i. Finally, a positive constant C as a default wheel speed is added. $\mathbf{N_l}$ and $\mathbf{N_r}$ are set of channel number of left- and right-output neurons. Here, as ω_i is a negative value and C is a positive value, the robot moves forward when the output neurons are not active. As the activities of the output neurons increase, the speed of the forward movement decreases and finally the robot moves backwards. As the two wheels of the robot are independent, the robot can also turn.

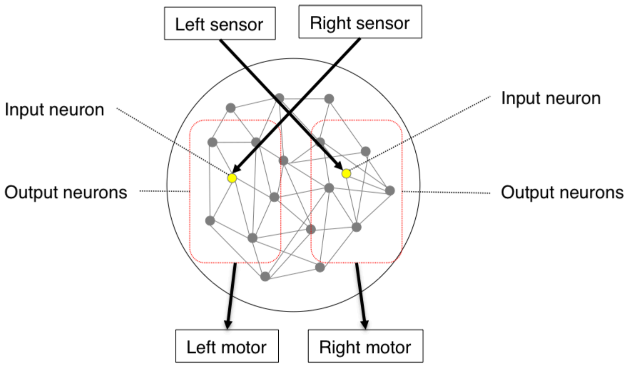


Figure 3: Sensorimotor mapping between a robot and cultured neuronal cells. Two electrodes on the CMOS array are selected as input neurons and connected to the distance sensor of the mobile robot. Twenty electrodes are selected as output neurons and connected to the left- and right-wheel speed of the mobile robot.

Results

A robot was placed in the 60 cm 60 cm arena (Figure 4), and both the neural activities and the behavior of the robot (1 h) were recorded using cultured neuronal cells under two different conditions (Chip#1[DIV 28] and Chip#2[DIV 38]), where Chip#1 is the neural assembly of 28 days and Chip#2 is 28 days after sowing. In previous research, we studied the difference in neural behavior stemming from the different conditions (Matsuda et al. (2013)). The neural spiking patterns were recorded in the pre- (1 h) and post- (1 h) duration of the coupling experiment between the robot and the neural cells. A video recording was used for tracking the trajectories, and we used the open-source software SwisTrack (Correll et al. (2006)) for tracking the trajectories. We also recorded the right- and left-sensor input values of the robot.

In the following section, we show the results of whether wall-avoidance behavior is improved autonomously by analyzing the trajectory and sensor value of the robot. We then provide the analysis of neural connectivity for supporting the improvement of wall-avoidance behavior.

Figure 4: Experiment environment. The robot is placed in the square arena (60cm60cm).

Evaluation of wall-avoidance behavior

We focused on whether the mobile robot could improve wall-avoidance behavior autonomously. Figure 5 shows a trajectory of the robot in the experiment with Chip#1. Qualitatively it appears that the activity pattern changed, which we took as a sign of the modifiability of the networks.

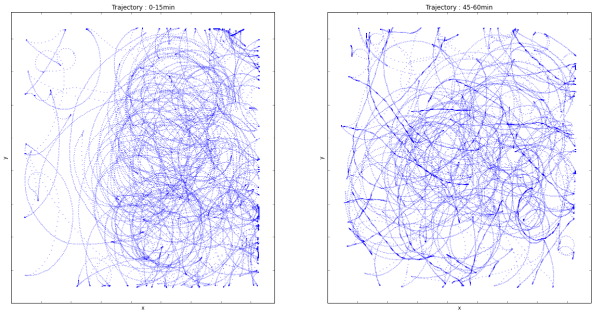


Figure 5: Trajectory of a robot in the Chip#1 experiment. The left panel shows the trajectory of the first 15 min in the robot experiment; the right panel shows the trajectory of the last 15 min.

Although a quantitative evaluation of the behavior is needed, it is difficult to track the direction of the robot from the video data. Thus, we used wall-collision time for evaluating the wall-avoidance behavior. This is defined as the duration between the time in which a sensory input value exceeds the upper threshold and the time in which the sensory input value is lower than the lower threshold (Figure 6). When the robot collides with a wall or stands close to it, the sensor becomes activated, otherwise it receives a weaker signal. Therefore if the estimated wall-collision time become lower, we can conclude that the wall-avoidance behavior was enhanced.

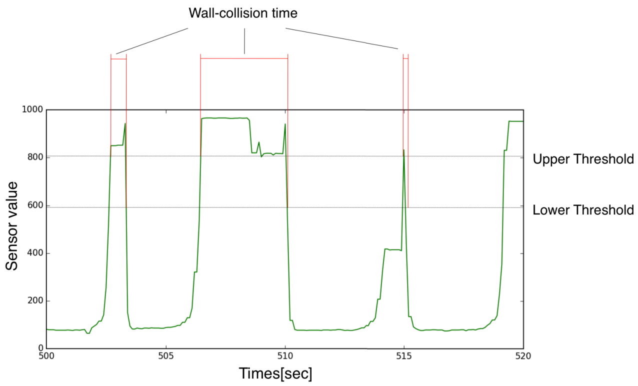
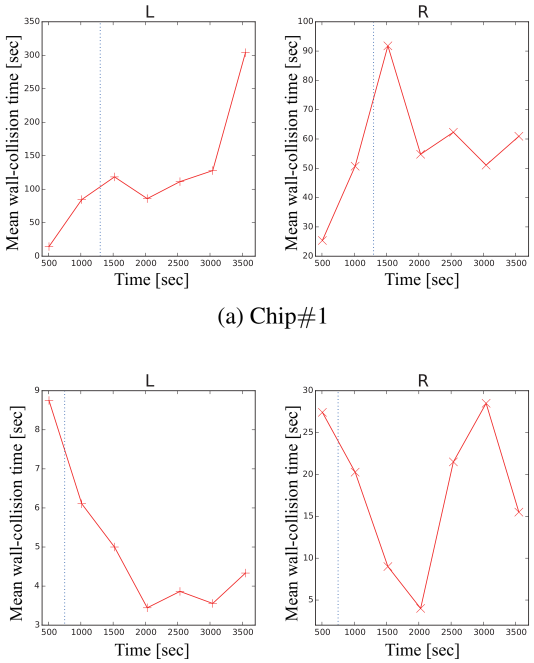


Figure 6: Definition of the estimated wall-collision time. The wall-collision time is defined as the duration between the time in which a sensor-input value exceeds an upper threshold and the time in which a sensor-input value is lower than the lower threshold.

Figure 7 shows the time series of the estimated wall-collision time. Here, relaxation time means the duration for stabilizing the number of stimulation-induced spikes of input neurons. A stimulation-induced spike is defined as the number of spikes of input neurons within 100 ms after each stimulation. Figure 8 shows the time series of the number of stimulation-induced spikes, which stabilized at nearly 1,300 s. Thus, in this case, relaxation time was set to 1300 s. Based on the time series of the estimated wall-collision time, the wall-avoidance behavior was not totally improved, yet the estimated wall-collision time of at least either the right or left sensor gradually decreased. In Figure 7(a), the estimated wall-collision time of the right-sensor input gradually decreased, and in Figure 7(b), that of the left-sensor input gradually decreased, indicating a partial improvement in the wall-avoidance behavior. We can therefore conclude that the LSA principle is working in this setup.

Analyzing dynamics of functional connectivity

We also analyzed the change in functional connectivity. A correlation between a pair of neurons may not represent a physical connection but a functional connection. Several methods are used for estimating the strength of a functional connection between neurons, including mutual information and transfer entropy (Schreiber (2000), Matsuda et al. (2013)). In this study, we used conditional firing probability (CFP) for detecting functional connectivity by using the cross-correlation between neural states (le Feber et al., 2007). This is defined as follows:



(a) Chip#1

(b) Chip#2

Figure 7: Estimated wall-collision time. (a) Results of Chip#1 and lower figures. (b) Results of Chip#2. Each left figure shows the time series of the wall-collision time estimated from the left-sensor input of the robot. Each right figure shows the time series of the wall-collision time estimated from the right-sensor input of the robot. The blue dotted line represents the relaxation time for stabilizing the simulation-induced spike of the input neuron.

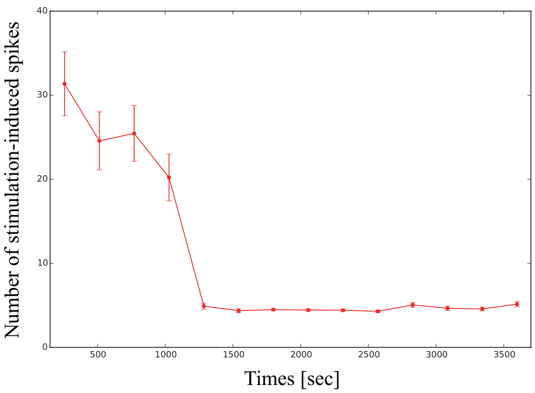


Figure 8: Time series of the number of stimulation-induced spikes of the input neuron in Chip#1. A stimulation-induced spike is defined as the number of spikes of input neurons within 100 ms after each stimulation. In this case, the stimulation-induced spike stabilized at around 1300 s, indicating the total relaxation time.

$$\mathrm{CFP}_{i,j}(\tau) = \frac{\sum_t X_i(t)X_j(t+\tau)}{\sum_t X_i(t)} \quad (0 < \tau \leq 500msec)$$

$X_{i,j}$ are binary arrays of firing in which 0 represents no firing at electrode $i; j$ and 1 represent at least one firing at electrode $i; j$ within the given time window. Thus, $\mathrm{CFP}_{i,j}(\tau)$ represents the firing rate at electrode j at a delay time τ $(0 < \tau \leq 500ms)$ after the firing at electrode i divided by the total number of firings at electrode i. The interval of τ is 1 ms.

Figure 9 shows an example of a CFP curve fitted by the following equation using the nonlinear least squares method to minimize the mean squared error.

$$\mathrm{CFP}^{fit}_{i,j}(\tau) = \frac{M_{i,j}}{1 + (\frac{\tau - T_{i,j}}{w_{i,j}})} + \mathrm{offset}_{i,j}$$

$M_{i,j}$ represents the maximum value above the offset, and $T_{i,j}$ represents the time at which the CFP^{fit} function reaches the maximum value. The shape of the curve is determined by the parameter $\omega_{i,j}$. The $\mathrm{offset}_{i,j}$ reflects unrelated background activity. In this study, if $M_{i,j}$ is two times greater than the $\mathrm{offset}_{i,j}$ level and $T_{i,j}$ is larger than zero and does not exceed 250 ms, $T_{i,j}$ is regarded as a functional connection between electrode $i; j$ and $M_{i,j}$ is regarded as the estimated strength of functionally connected electrodes $i; j$.

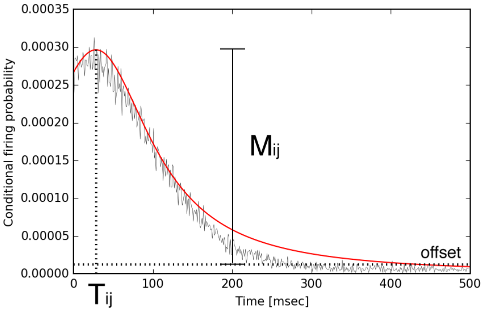
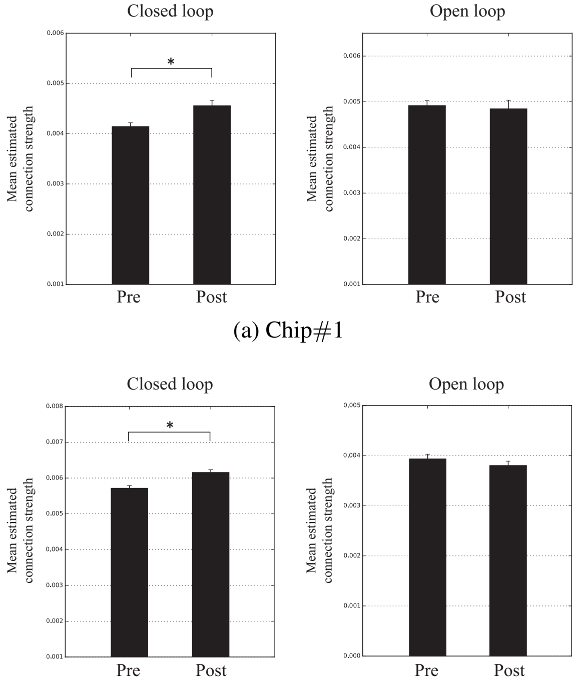


Figure 9: An example of the conditional firing probability (CFP). A gray line represents the CFP curve. A red line represents the fitted curve to the CFP curve. $M_{i,j}$ represents the maximum value above the offset, $T_{i,j}$ represents the time at which the CFP^{fit} function reaches the maximum value, and offset represents the unrelated background activity.

Using this CFP method, we calculated the strength of functional connectivities between the electrodes and compared the pre-experiment and the post-experiment values to evaluate the changes in functional connectivity. We also compared it with the result of open loop experiment. Here open loop experiment means the experiment in which the stimulation is sent to the input neuron at the same timing with that of robot experiment, but the cultured neuronal cells is not connected to the robot, then there is no feedback to the stimulation pattern from neural activity. The open loop experiments was conducted using same chip with the closed-loop experiments. Results showed that the mean strength of the functional connection in the post-experiment was significantly greater than that of the pre-experiment ($p < 0.05$, Wilcoxon rank-sum test) in a closed-loop experiment but that there was no significant difference in the results of the open loop experiment where (Figure 10).



(a) Chip#1

(b) Chip#2

Figure 10: Comparison of the estimated strength of a functional connection. Upper figures are the results of Chip#1, and lower figures show the results of Chip#2. Each left figure is a closed-loop experiment (robot experiment), and each right figure is an open loop experiment. In the closed-loop experiment, the estimated strength of the functional connection in the post-experiment is significantly greater than that of the pre-experiment (*$p < 0.05$, Wilcoxon rank-sum test), but there is no significant difference in the results of the open loop experiment.

The results of the closed-loop experiment showed that the functional connectivity between the input neuron and the output neuron increased as did the synchronization between the two neuronal states. An easier way to improve wall-avoidance behavior in the experimental setup is to have many output neurons fire at approximately the same time after the stimuli. If this happens, we can conclude that wall-avoidance behavior by the LSA principle has improved (see also Sinapayen et al. (2015), for a discussion of LSA with artificial neural network experiments)

Discussion

We usually train a robot to maximize a reward function either by on-line or off-line learning. Without any reward function, it is difficult to self-organize sense making or a goal-oriented behavior. We scarcely have such autonomous robots; the present study challenges this problem.

Hanczyc and Ikegami (2010) demonstrated a simple but spontaneous sense-making behavior with a simple chemistry. Water at a high pH reacting with oleic anhydride generates self-moving droplets, which maintain the reaction on its surface, sustaining its self-mobility (Toyota et al. (2009); Hanczyc and Ikegami (2010)). As a result, a droplet climbs up the pH gradient when the environmental pH is 12 and turns away from it when pH is 10. This pseudo chemotaxis of oil droplets was easily introduced by an internal chemical reaction plus embodiment. This behavior pattern feeds back into a chemical reaction of a droplet to sustain its activity.

We aimed to demonstrate that a tendency to escape from a stimulus generates autonomous motion and in the end leads to a goal-oriented behavior. We call this LSA. We aimed to evaluate this learning hypothesis with a simple robot experiment. In this study, we implemented a closed-loop system for connecting a robot and cultured neuronal cells and conducted the experiment using this system. The results showed that wall-avoidance behavior is partially improved, and we consider that such sense-making behavior can be enhanced autonomously even though there are no explicit external rewards. We argue that this emerging sense-making behavior is an outcome of integrating embodiment, the environment, and adequate sensors. In this paper we focused on wall-avoidance behavior; however, we hypothesize that other sense-making behaviors can emerge from other couplings of embodiment and environment.

Previously, Di Paolo (2000), Iizuka and Di Paolo (2007) and Di Paolo and Iizuka (2008) studied the emergence of sense-making behavior as an outcome of neural homeostasis. Ikegami (2013) developed an experimental art system (called “Mind Time Machine” [MTM]) consisting of three screens and 15 video cameras. In MTM, each camera iteratively projects images taken from those screens over 3 months. Those in-take images are edited and modified by the underlying artificial neural dynamics. The complexity of the environment and the stored image memories of MTM create a life-like impression for the people who interact with MTM. MTM created a sense-making behavior without any reward function.

The emergence of sense-making behavior is and has been a major theme for artificial life and should be explored further. A drawback of the robot-neural platform is that it uses the entire network for making one style of behavior. This can be improved by i) promoting module networks for learning and ii) restoring and retrieving many memories. However, we primarily need more studies to show how LSA works practically for evolving sense-making behavior. Furthermore, we hypothesize such an LSA principle can be generalized to not only cultured neural networks but also to an artificial neural network and to any system with other elements that include modifiability and stability, for example, a microorganism that has no nervous system.

Acknowledgements

The high-density CMOS array used in this study was provided by Prof. Andreas Hierlemann, ETH Zürich. This work was supported by a Grant-in-Aid for Scientific Research (Studies on Homeo-Dynamics with Cultivated Neural Circuits and Embodied Artificial Neural Networks; 24300080).

References

Ashby, W. R. (1960). *No TitleDesign for a Brain: The Origin of Adaptive Behavior*. Chapman and Hall, 2 edition.

Bakkum, D. J., Chao, Z. C., and Potter, S. M. (2008). Spatio-temporal electrical stimuli shape behavior of an embodied cortical network in a goal-directed learning task. *Journal of neural engineering*, 5(3):310–23.

Correll, N., Sempo, G., De Meneses, Y. L., Halloy, J., Deneubourg, J. L., and Martinoli, A. (2006). SwisTrack: A tracking tool for multi-unit robotic and biological systems. In *IEEE International Conference on Intelligent Robots and Systems*, pages 2185–2191. Ieee.

DeMarse, T. and Dockendorf, K. (2005). Adaptive flight control with living neuronal networks on microelectrode arrays. *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, 3:1548–1551.

Di Paolo, E. A. (2000). Homeostatic adaptation to inversion of the visual field and other sensorimotor disruptions. *From Animals to Animats VI: Proceedings of the 6th International Conference on Simulation of Adaptive Behavior*, pages 440–449.

Di Paolo, E. A. and Iizuka, H. (2008). How (not) to model autonomous behaviour. *BioSystems*, 91:409–423.

Eytan, D. and Marom, S. (2006). Dynamics and effective topology underlying synchronization in networks of cortical neurons. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 26(33):8465–76.

Frey, U., Sedivy, J., Heer, F., Pedron, R., Ballini, M., Mueller, J., Bakkum, D., Hafizovic, S., Faraci, F. D., Greve, F., Kirstein, K.-U., and Hierlemann, A. (2010). Switch-Matrix-Based High-Density Microelectrode Array in CMOS Technology. *IEEE Journal of Solid-State Circuits*, 45(2):467–482.

Hanczyc, M. M. and Ikegami, T. (2010). Chemical basis for minimal cognition. *Artificial life*, 16(3):233–43.

Hayashi, I., Tokuda, M., Kiyohara, A., Taguchi, T., and Kudoh, S. N. (2011). Biomodeling System: Interactive Connection between Cultured Neuronal Network and Moving Robot Using Fuzzy Interface. *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics*, 23(5):761–772.

Iizuka, H. and Di Paolo, E. a. (2007). Toward Spinozist Robotics: Exploring the Minimal Dynamics of Behavioral Preference. *Adaptive Behavior*, 15(4):359–376.

Ikegami, T. (2013). A Design for Living Technology : Experiments with the Mind Time Machine. *Artificial Life*, 400:387–400.

Madhavan, R., Chao, Z. C., and Potter, S. M. (2007). Plasticity of recurring spatiotemporal activity patterns in cortical networks. *Physical biology*, 4(3):181–93.

Matsuda, E., Mita, T., Hubert, J., Bakkum, D., Frey, U., Hierlemann, A., Takahashi, H., and Ikegami, T. (2013). Analysis of neuronal cells of dissociated primary culture on high-density CMOS electrode array. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 2013:1045–8.

Novellino, A., D'Angelo, P., Cozzi, L., Chiappalone, M., Sanguineti, V., and Martinoia, S. (2007). Connecting neurons to a mobile robot: an in vitro bidirectional neural interface. *Computational intelligence and neuroscience*, 2007:12725.

Pizzi, R. M. R., Rossetti, D., Cino, G., Marino, D., A L Vescovi, and Baer, W. (2009). A cultured human neural network operates a robotic actuator. *Bio Systems*, 95(2):137–44.

Potter, S. M. and DeMarse, T. B. (2001). A new approach to neural cell culture for long-term studies. *Journal of Neuroscience Methods*, 110(1-2):17–24.

Schreiber, T. (2000). Measuring Information Transfer. *Physical Review Letters*, 85(2):461–464.

Shahaf, G. and Marom, S. (2001). Learning in networks of cortical neurons. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 21(22):8782–8.

Sinapayen, L., Masumori, A., Virgo, N., and Ikegami, T. (2015). Learning by Stimulation Avoidance as a Primary Principle of Spiking Neural Networks Dynamics. In *The 13th European Conference on Artificial Life (ECAL 2015), (in press)*.

Toyota, T., Maru, N., Hanczyc, M. M., Ikegami, T., and Sugawara, T. (2009). Self-propelled oil droplets consuming "Fuel" surfactant. *Journal of the American Chemical Society*, 131:5012–5013.

Wagenaar, D. a., Madhavan, R., Pine, J., and Potter, S. M. (2005). Controlling bursting in cortical cultures with closed-loop multi-electrode stimulation. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 25(3):680–8.

Wagenaar, D. A. and Potter, S. M. (2002). Real-time multi-channel stimulus artifact suppression by local curve fitting. *Journal of neuroscience methods*, 120(2):113–20.

Warwick, K. (2010). Implications and consequences of robots with biological brains. *Ethics and Information Technology*, 12(3):223–234.

Omnigram Explorer: A Simple Interactive Tool for the Initial Exploration of Complex Systems

Tim Taylor, Alan Dorin and Kevin Korb

Faculty of Information Technology, Monash University, VIC 3800, Australia
tim@tim-taylor.com alan.dorin@monash.edu kbkorb@gmail.com

Abstract

We describe the design of Omnigram Explorer (OMG), an open-source tool for the interactive exploration of relationships between variables in a complex system. OMG is designed to help researchers gain a holistic, qualitative understanding of the relationships between variables in their data at a preliminary stage of analysis; such exploration might highlight interactions that warrant further quantitative investigation using other tools. We illustrate OMG's use on real-world data, and also describe its potential as a tool for communication to non-specialists.

Introduction

When working with models of complex systems, visualisation tools can be invaluable in helping researchers gain an understanding of the system's behaviour. Visualisation can be used at all stages of the research process, from the earliest stages of design through to the eventual communication of results to a variety of different audiences (Dorin and Geard, 2014; Grimm, 2002).

The work reported in this paper was developed in the context of a collaboration between the authors and a group of epidemiologists at the University of Melbourne.[1] The epidemiologists use a variety of different modelling techniques in their work, ranging from agent-based models and other simulations to more traditional mathematical approaches. Their models often involve several dozen independent and dependent variables.

We wished to develop an interactive tool that would allow our colleagues to gain a quick qualitative understanding of their models at the initial stages of analysis, and highlight features to be investigated in more detail. The resulting tool, named Omnigram Explorer (which we abbreviate as OMG), is described in this paper. In the following sections we review relevant previous work, describe the principles that drove OMG's design, describe the tool's main features, and give examples of using OMG on real-world data.[2]

Omnigram Explorer is a free, open-source tool developed in Processing.[3] The source code, binary executables (for Windows, Mac and Linux), documentation and related materials are available at `http://www.tim-taylor.com/omnigram`.

Previous work

Traditional approaches to visualising the relationships among multiple variables include the Scatter Plot Matrix (SPLOM) and Parallel Coordinates (Heer et al., 2010). These are widely used, although neither is without problems. For example, SPLOM visualisations focus on pairwise relationships between variables (Kosara et al., 2003), and the effectiveness of Parallel Coordinates can greatly depend on factors such as the linear order in which data dimensions are plotted (Zhang et al., 2012).

In addition to representation, *interaction* is an increasingly important aspect of information visualisation systems (Ward and Yang, 2004; Yi et al., 2007). For the kinds of visualisation problems we wished to address in our project, relevant and interesting early work was produced by Spence and Tweedie (1998). They developed a tool called *Attribute Explorer* in which multivariate data was presented as a set of histograms, one for each variable. Each histogram displayed the distribution of values in the data-set for its associated variable. The user could select subsets of data by adjusting a slider under a histogram. The subset of data points so selected was represented in the other histograms using a specific highlight colour (a technique known as *linking and brushing*).

The tool described in the current paper, Omnigram Explorer, took inspiration from Spence and Tweedie's work as a starting point. We added a variety of novel and principled extensions (as described in the following sections), and have made it available as a free, open-source tool.

Design principles

An effective data visualisation should provide an intuitive way for the user to gain insight into the organisation of that

[1] See Acknowledgements section for details.

[2] Various supplementary materials are indicated in the paper. These are available at `http://www.tim-taylor.com/omnigram/ecal2015/`.

[3] `http://processing.org`

data. Effective interaction in an information visualisation should allow the user to develop a mental model of correlations and relationships in the data (Kosara et al., 2003).

A guiding principle behind the development of Omnigram Explorer was to leverage properties of the human visual system to allow complex information to be easily processed by the user. In particular, we used several Gestalt grouping principles[4] in the design of the system to allow it to convey a large amount of information about correlations in the data in a deceptively simple way, and to allow users to focus on some variables and not on others. We will expand upon these features in the following sections.

Features

General Usage

OMG is a tool for visualising pre-recorded data of parameter settings and output values from a set of runs of a target model.

At start-up, the user is prompted to select a model definition file that describes the system's variables and also points to a data file. The data file contains the samples of values for each variable. OMG loads this data (or a random subset of it if so instructed) and presents it to the user in graphical form for interactive exploration. OMG is agnostic about the particular form of the system and its variables: variables may be designated as inputs or outputs if desired, but this is not required. The model definition file may also contain information about causal relationships between variables, which can then be visualised to provide hints to the OMG user of interesting relationships to explore. Information about causal relationships might have been generated from existing knowledge of the system or from tools such as Bayesian causal network discovery software.[5]

As described in the following sections, OMG provides various modes of interaction that allow the user to explore the relationships in the system in different ways.

User Interface and Nodes

A general view of the OMG interface is shown in Figure 1.[6]

The most important component of the user interface is the *node*, which is an interactive, graphical representation of the data associated with a particular variable in the system. The detailed user interface of a single node is shown in Figure 2. A node displays a histogram showing the distribution of samples of a particular variable observed in the data read in from the data file.

The node's histogram always shows the distribution of *all* samples read in from the file. However, a subset of these samples may be highlighted in different colours, either (for focus nodes) to indicate that they lie within a range of values selected with the range selector widget by the user, or (for non-focus nodes) to represent brushing in response to focus nodes (see later for further details of focus nodes and brushing). A small circle is drawn below the histogram to indicate the position of the median (or mean) value of the selected samples.

By showing the selected samples highlighted against non-selected samples, the node therefore provides *context* and *guidance* for later exploration; this was one of the guiding principles of *Attribute Explorer* (Spence and Tweedie, 1998). We use the term *omnigram* to refer to the simultaneous visualisation of histograms of all of a system's parameters; an omnigram extends these principles of context and guidance from a single parameter to the whole system.

Modes of Interaction

There are four basic modes of interaction in OMG: *Single Node Brushing*, *Multi Node Brushing*, *Omnibrushing* and *Sample View*.

Single Node Brushing In this mode, only one node can have the focus at any one time. Clicking on the histogram area of a node gives it the focus (shown by the red focus indicator in the top left of the node), and removes the focus from any other node.

The range selector of the focus node can be adjusted to select a subset of samples from the overall distribution (i.e. a subset of bins from the histogram). The handles at each end of the selector can be dragged left or right to change the upper and lower limits of the selected range, and the main selector bar can be dragged left or right to shift the whole range up or down. The bins within the selected range are shown in dark red, and the other bins in the focus node are shown in white.

When a range of samples has been selected in the focus node in this way, all of the other nodes are updated to show where the same samples lie in the distributions of the other variables in the model. The matching samples are shown in dark blue. This is OMG's implementation of the *linking and brushing* technique mentioned earlier.

The real power of linking and brushing in OMG becomes apparent when you interactively change the range selection in the focus node, and watch the resulting changes in the other nodes. In particular, selecting a fairly small subset of values with the range selector in the focus node (i.e. having a short range selector bar), dragging the bar from left to right and back, and watching how the change in values of the focus node is associated with changes in the other nodes, is a very effective technique.

The human visual system is very attuned to noticing multiple objects moving in the same direction (the Gestalt principles of Continuity and Common Fate). OMG makes use of

[4] See, e.g., (Wolfe et al., 2009).

[5] E.g. CaMML (Korb and Nicholson, 2011).

[6] The screen-shots in this section show OMG using data from the standard Auto MPG data-set downloaded from the UCI Machine Learning Repository (Lichman, 2013) and with causal links generated by CaMML (Korb and Nicholson, 2011).

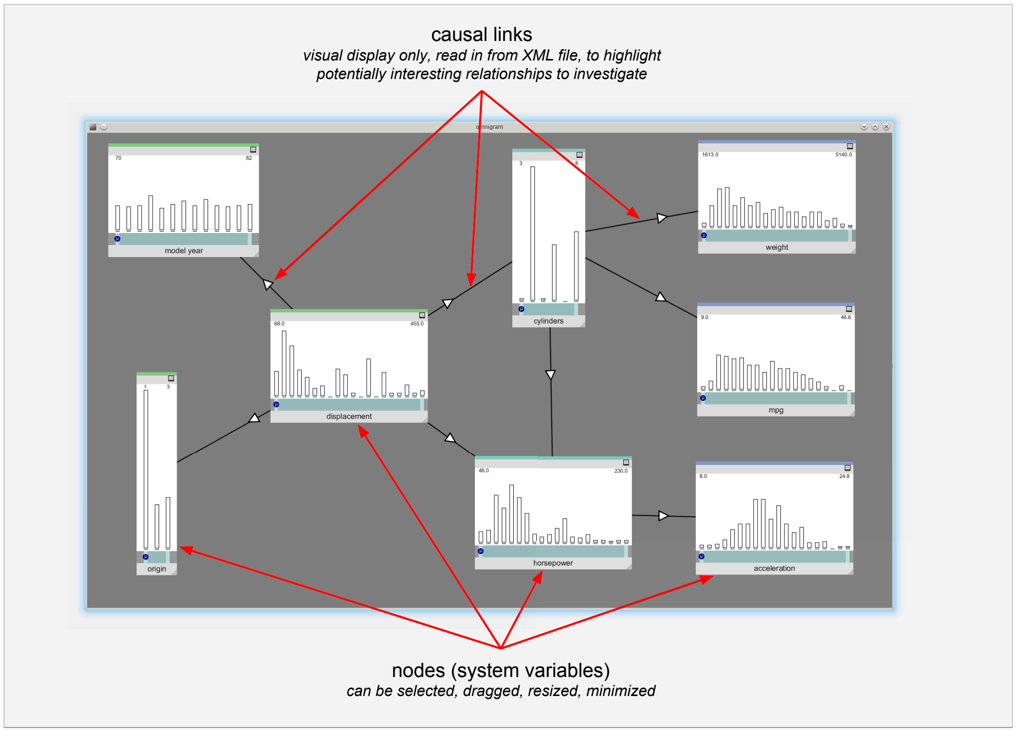

causal links
visual display only, read in from XML file, to highlight potentially interesting relationships to investigate
omnigram
model year
cylinders
weight
displacement
mpg
origin
horsepower
acceleration
nodes (system variables)
can be selected, dragged, resized, minimized

Figure 1: The Omnigram Explorer user interface.

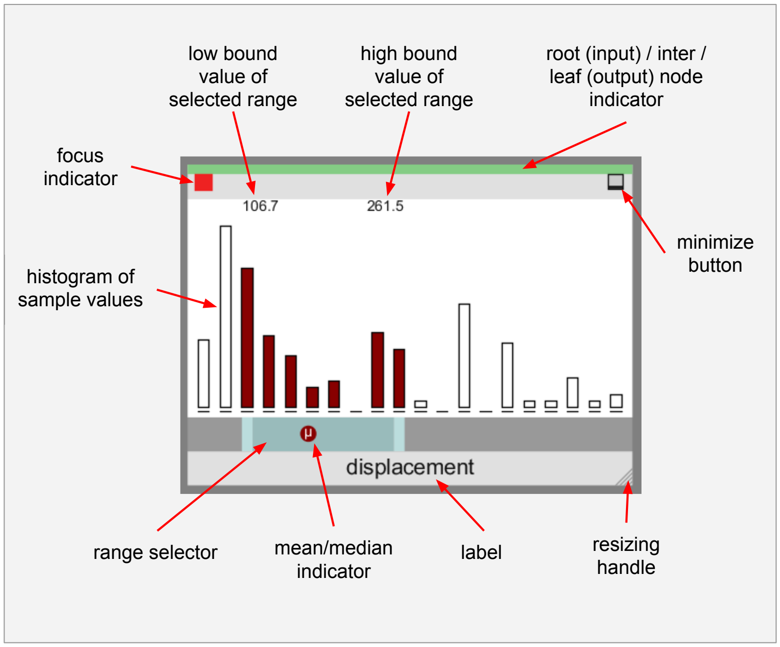

low bound value of selected range
high bound value of selected range
root (input) / inter / leaf (output) node indicator
focus indicator
106.7
261.5
histogram of sample values
minimize button
μ
displacement
range selector
mean/median indicator
label
resizing handle

Figure 2: Detail of user interface for a Node.

this fact to allow the user to easily spot correlations between many different variables in *Single Node Brushing* mode. Because these principles rely on movement, it is difficult to convey the power of this technique in writing or still images. By interacting with the system (moving the range selector bar) and watching how the selected set of samples change in each histogram as a result, it is very easy to see how changes in one variable are correlated to changes in all of the other variables in the system. Feedback from informal focus group meetings[7] suggests that this is effective even with 30–40 nodes on-screen at once. A demonstration video is provided in the supplementary materials.

Multi Node Brushing *Multi Node Brushing* is an extension to *Single Node Brushing* that provides information about sensitivity of the model. An example screen-shot of *Multi Node Brushing* mode is shown in Figure 3.

In this mode, more than one node can have the focus at the same time. In the example shown in the figure, the three leftmost nodes all have the focus (indicated by the red squares at the top left of each node). The range selectors in each of the focus nodes can be adjusted just like in *Single Node Brushing* mode, to select a subset of samples.

The information displayed in non-focus nodes is more detailed than in *Single Node Brushing* mode. As before, dark blue is used to indicate the location of samples that have been selected by the range selection in the focus nodes. In *Multi Node Brushing*, OMG looks at the conjunction of the range selections made on the focus nodes: to take an example from Figure 3, a blue patch on a histogram bin in the *horsepower* node indicates that there is a sample that has that horsepower value that also lies within the selected range of *model year* and lies within the selected range of *displacement* and lies within the selected range of *origin*.

You will notice from the figure that non-focus nodes also show other colours. A light green patch indicates that there is a sample that has that value that also lies within the selected range of values for *all but one* of the focus nodes. So, in this example, a light green patch on a histogram bin in the *horsepower* node might indicate that there is a sample that has that horsepower value that also lies within the selected range of *model year* and lies within the selected range of *displacement*, but not within the selected range of *origin*. It might also indicate that the origin and model year ranges were respected, but not the displacement, etc. That is, the light green colour indicates that one of the focus node ranges was not respected, but it does not tell you which one.

Similarly, a light red patch indicates that two of the focus node ranges were not respected, and white indicates that three or more were not respected.

The colour therefore gives some indication of the sensitivity of the model to the value ranges chosen. For example, the presence of a light green patch in a non-focus node indicates that by increasing the selected range of just one of the focus nodes, the patch can be turned blue, i.e. it can be made to satisfy the constraints on all of the focus nodes by loosening a single constraint.

Omnibrushing *Omnibrushing* mode works in a somewhat different way to *Single Node Brushing* and *Multi Node Brushing*. Like in *Single Node Brushing*, only one node can have the focus at a time in *Omnibrushing* mode. When a node receives the focus, a different colour is assigned to each of that node's bins. The hue of each bin changes steadily from the leftmost bin to the rightmost bin, with reds on the left changing to greens and blues on the right.

The colours in each of the non-focus nodes are updated to reflect those of the focus node. For each bin in a histogram of a non-focus node, a fraction of the area of the bin is given the same colour as each of the bins in the focus node according to the fraction of samples in that non-focus bin that belong to the bin in the focus node corresponding to that colour. An example of *Omnibrushing* mode is shown in Figure 4; for further examples, see the Example Usage section and Figure 6 below.

Thus, *Omnibrushing* mode can be thought of as a composite *Single Node Brushing* mode, where each bin in the focus node has been given a different colour, and the corresponding brushing of all other nodes has then been superimposed into a single representation.

Sample View *Sample View* mode provides a different way to visualise the data. The overall distribution of values for each node is shown, as with the other modes, but these are shown partially faded out in the background of each node, and are provided just as a reminder of the overall shape of the observed data. The real point of interest in this mode is the display of individual samples from the data file. Each sample is shown as a small coloured circle placed in the position of the histogram bin corresponding to the sample value of that variable. An example screen from *Sample View* mode is shown in Figure 5.

At any one time, a fixed number of samples is shown. By default, OMG will automatically cycle at a moderate speed through different selections of samples from the data. At each step in the cycle, one sample (the one that has been displayed for the longest time) is removed from the currently displayed set, and a new sample replaces it. The user can interactively increase or decrease both the number of samples shown at a time and the speed of cycling. The user may also choose to step forwards and backwards through the different sample selections manually rather than having them updated automatically.

In *Sample View* mode, only a single node can have the focus at any one time. By default, the samples are coloured according to which bin they belong to in the focus node (for

[7] These meetings comprised eight participants (including the software author). More rigorous tests are planned in future work.

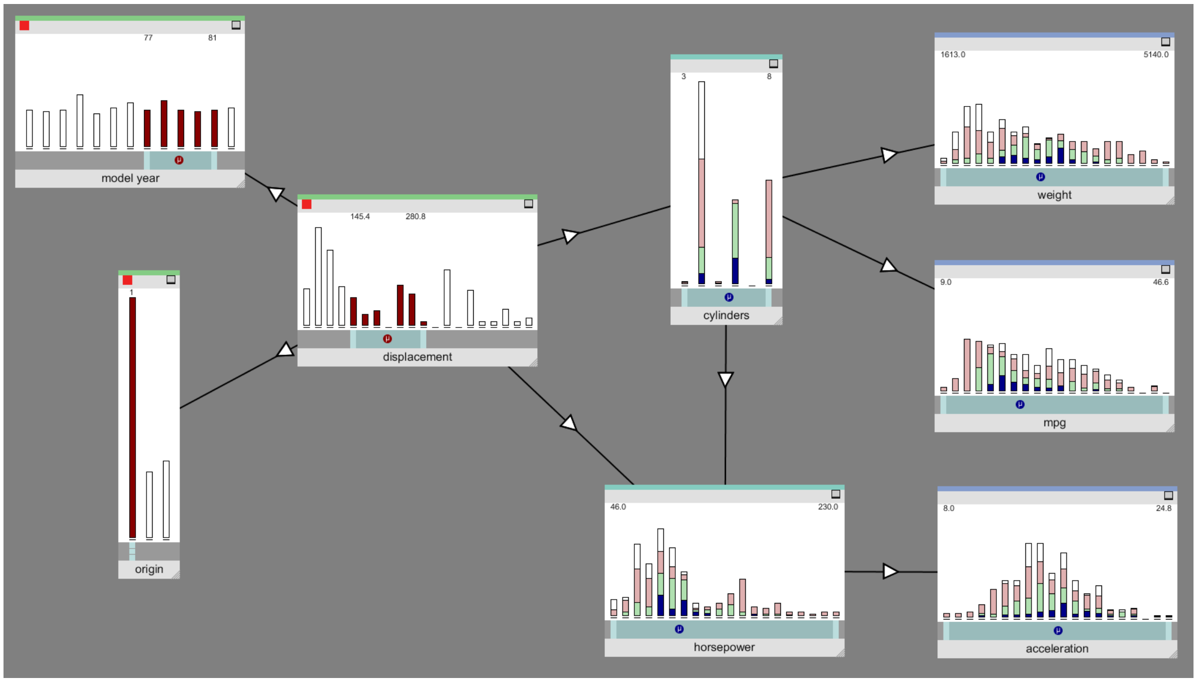


Figure 3: Multi Node Brushing mode (focus nodes: model year, origin, displacement)

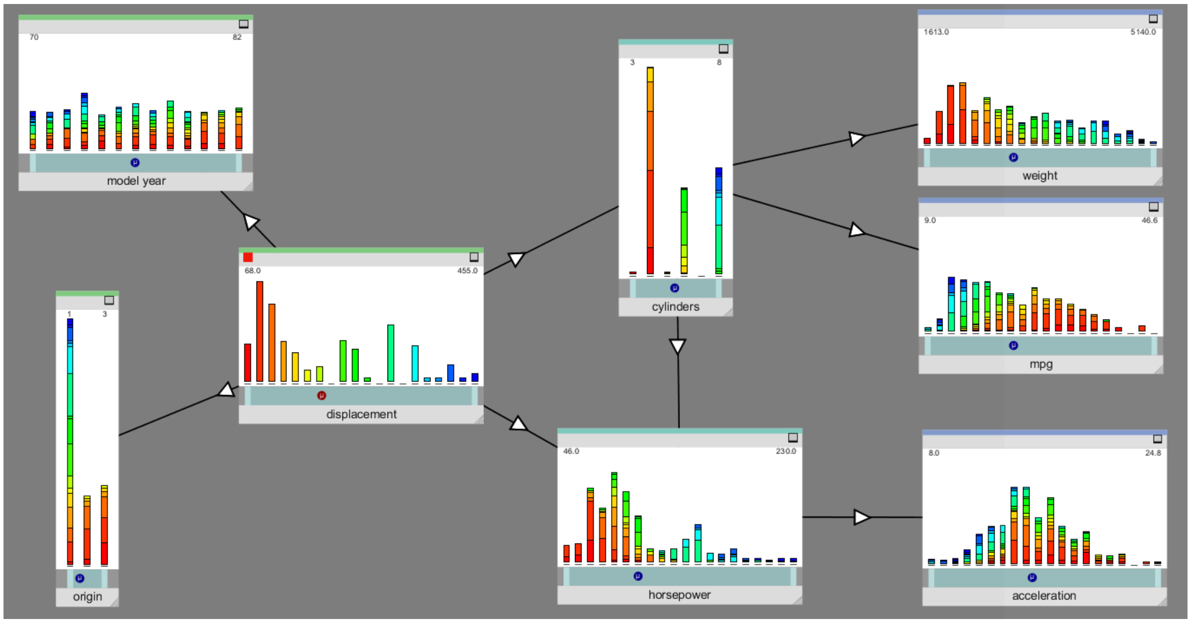


Figure 4: Omnibrushing mode (focus node: displacement)

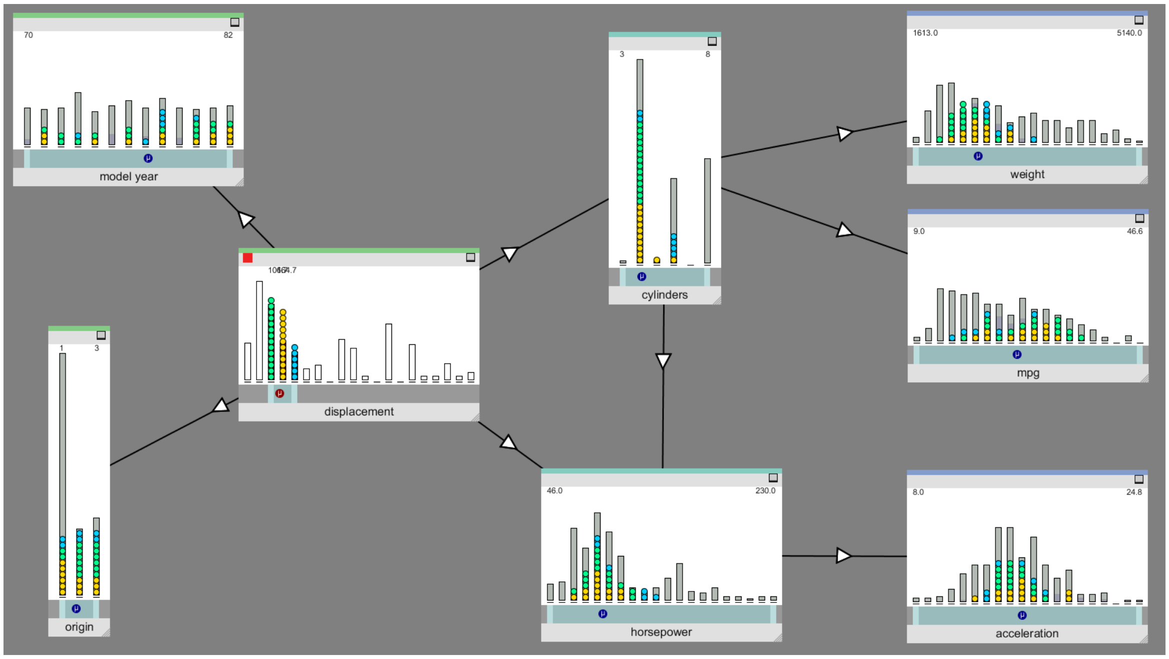


Figure 5: Sample View mode (focus node: displacement)

an example, see Figure 5, where the focus node is the *displacement* node, indicated by the red focus indicator in the top left of the node). The user can quickly see how these samples are distributed in the other nodes by looking at the distribution of colours.

If preferred, users can select a colouring scheme in which samples are coloured at random, irrespective of their bin in the focus mode. This scheme is most useful for studying a small number of samples. The colours allow the user to determine, for each coloured sample in the focus node, the exact bins into which they fall for all other nodes.

Other features

OMG has various other features that we will not describe in detail here. These include the ability to interactively create *brush links* between nodes in *Multi Node Brushing* mode, so that when the range selector on one focus node is moved, the range selector on another linked focus node also moves in a specified way. This allows the user to interactively explore complex relationships between multiple variables.

As mentioned before, the model definition file may also contain information about causal links between nodes, and OMG can display these as arrows between nodes to provide hints about relationships that might be worth exploring.

The initial placement of nodes on the screen is done automatically, and the input file may contain hints about which nodes should be placed where. But regardless of initial placement, all nodes can be dragged to new positions interactively by the user. One situation in which this can be particularly useful is as follows: if nodes have been identified that behave in a similar way (e.g. when dragging the range selector of the focus node), those nodes can be grouped together by dragging them to one part of the screen. The user can then start to partition a complex model into subsets of similarly performing variables by spatially grouping the nodes on screen. Placing similarly-behaving nodes in close spatial proximity allows the user to perceive them as a single group (the Gestalt principle of Proximity), thereby simplifying the process of mental model formation. Other methods to increase or decrease the focus on specific nodes include the ability to interactively resize a node, and to minimise it so that the histogram is completely hidden from view.

Example Usage

As an introductory example of using OMG on real data, we will briefly discuss its use on the standard Heart Disease data-set available from the UCI Machine Learning Repository. We use the reduced Cleveland subset of the data, comprising 297 complete samples, each with 13 input attributes and one disease diagnosis attribute to be predicted. See Table 1 for a summary and (Lichman, 2013) for full details.

The data was loaded into OMG. In *Single Node Brushing* mode, the diagnosis node (*num*) was given the focus, and the range selector bar reduced so that a single bin in the *num* histogram was selected. This narrow range selector bar was then dragged left and right to select different bins in the histogram in rapid succession. The resulting patterns of brushing in the other nodes gave a clear indication of which

Label	Description
age	Age in years
sex	Sex (0=female, 1=male)
cp	Chest pain type (1-4)
trestbps	Resting blood pressure (mmHg)
chol	Serum cholesterol (mg/dl)
fbs	Fasting blood sugar $>$ 120 mg/dl (0=no, 1=yes)
restecg	Resting ECG results (0-2)
thalach	Maximum heart rate achieved (bpm)
exang	Exercise induced angina (0=no, 1=yes)
oldpeak	ST depression induced by exercise
slope	Slope of peak exercise ST segment (1-3)
ca	Num major vessels coloured by fluoroscopy
thal	Summary of heart condition (3=normal)
num	Disease diagnosis (0=no presence)

Table 1: Heart Disease Data-set attributes

nodes were correlated with *num* (and the movement of the "median selected value" indicator under the histograms provided an additional indication). This process is illustrated in a video in the supplementary materials.

To quantify the effectiveness of this procedure, after performing this brushing technique for a few seconds, each node was given a qualitative rating of the degree to which it appeared to be correlated with the *num* node. Correlations were rated as *None*, *Low*, *Medium* and *High*. The results are shown in Table 2, along with the Pearson correlation coefficient of each attribute with *num*, calculated from the raw data. The qualitative assessments of correlation from OMG were then converted to numeric scores under the mapping {*None*=0, *Low*=1, *Medium*=2, *High*=3}, and the Pearson correlation coefficient between these scores and the calculated correlations was calculated. This came to 0.84, which demonstrates that the qualitative impression of the 13 correlations that can be gleaned from a few seconds of using OMG corresponds very well to the correlations calculated using Pearson's correlation coefficient.

A useful next step is to use *Omnibrushing* mode to gain another perspective on these relationships. A screen-shot of using OMG on this data in *Omnibrushing* mode, with the diagnosis node (*num*) selected as the focus node, is shown in Figure 6. Because *num* (the rightmost node in the figure) has the focus, each bin in that node is drawn using a different colour, ranging from reds on the left to blues on the right. The distributions of these samples in each of the other nodes is plotted using matching colours.

A wealth of information is revealed in Figure 6, but just a few points will be highlighted here. It is immediately apparent that this data shows that heart disease (indicated by *num*>0) is much more prevalent in males (*sex*=1) than females: look at the relative fraction of each bin in the *sex* node filled by healthy (red) samples. The prevalence of heart disease appears to be considerably higher in older people: the bins on the right-hand side of the *age* node, starting from the tallest bin (which corresponds to age 55), each show around 50% or more of samples with a disease diagnosis (non-red). On the other hand, the serum cholesterol level (*chol* node) does not appear to be strongly correlated, as each bin has roughly the same fraction of samples from each diagnosis.

Attribute	Apparent Correlation	Pearson Correlation
age	Medium	0.222
sex	Low	0.227
cp	High	0.404
trestbps	Medium	0.160
chol	None	0.066
fbs	None	0.049
restecg	Low	0.184
thalach	-Medium	-0.421
exang	Low	0.392
oldpeak	High	0.501
slope	Medium	0.375
ca	High	0.521
thal	High	0.513

Table 2: Qualitative judgement of apparent correlations of attributes with disease diagnosis using OMG in *Single Node Brushing* mode (column 2) and calculated Pearson correlation coefficients (column 3). See text for details.

These two initial investigations have suggested some relationships that might be worth exploring further. *Multi Node Brushing* mode can be used to begin to explore more complex relationships between multiple nodes. And *Sample View* mode can provide more fine-level investigation of patterns from individual samples in the data. Taken together, these investigations allow one to build up a useful qualitative overview of the data. These initial investigations can then suggest interesting questions for further quantitative analysis using other tools.

Other uses of OMG

Our colleagues at the University of Melbourne have started to use OMG to explore some of their data-sets. In addition to the kind of exploratory analysis described above, they have found it useful in other ways. Firstly, it is useful for quickly checking the independence of factors in a model. Secondly, it is suitable as a tool for communicating the behaviour of complex models to non-specialists such as government policy makers. Features that make it well suited for this task include the fact that data is presented as bar charts, which are easily understood. In addition, the scientists can deliver the OMG tool and allow non-specialists to interact with it using a previously generated data-set, thereby providing some constraints on the interactions (i.e. the non-specialists cannot break the model by doing something unexpected).

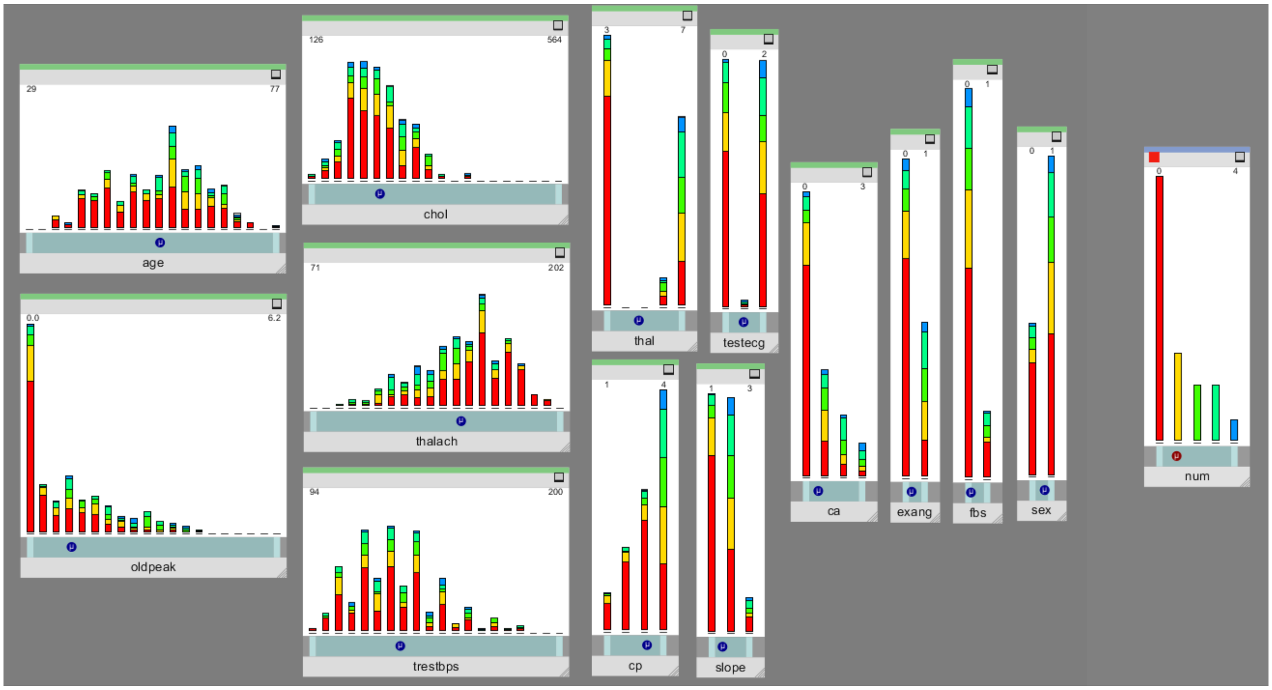


Figure 6: Example using Cleveland Heart Disease Data-set in *Omnibrushing* mode (focus node: num)

In future work we plan to extend OMG's capabilities in other ways. In particular, we are investigating connecting OMG to live processes in the form of simulations and Bayesian nets for live interaction and guided exploration.

Conclusion

We have described Omnigram Explorer, a tool for early-stage, qualitative exploration of data from simulations and other models of complex systems. The tool leverages properties of the human visual system to allow users to quickly spot relationships in the data through interactive manipulation. It has also been suggested that OMG has desirable properties for a tool for the communication of model dynamics to non-specialists. The tool is open-source, and we encourage readers to use it, and to extend it.

Acknowledgements

Omnigram Explorer was developed by the authors in collaboration with Jodie McVernon, James McCaw, Nicholas Geard and Patricia Campbell at the Melbourne School of Population and Global Health at the University of Melbourne. The project was funded by ARC grant number DP110101758.

References

Dorin, A. and Geard, N. (2014). The practice of agent-based model visualisation. *Artificial Life*, 20(2):271–289.

Grimm, V. (2002). Visual debugging: A way of analyzing, understanding and communicating bottom-up simulation models in ecology. *Natural Resource Modeling*, 15(1):23–38.

Heer, J., Bostock, M., and Ogievetsky, V. (2010). A tour through the visualization zoo. *Communications of the ACM*, 53(6):59–67.

Korb, K. B. and Nicholson, A. E. (2011). *Bayesian Artificial Intelligence*. CRC Press, 2nd edition.

Kosara, R., Hauser, H., and Gresh, D. L. (2003). An interaction view on information visualization. In *State-of-the-Art Proceedings of EUROGRAPHICS*, pages 123–137.

Lichman, M. (2013). UCI machine learning repository. http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences.

Spence, R. and Tweedie, L. (1998). The attribute explorer: information sythesis via exploration. *Interacting with Computers*, 11:137–146.

Ward, M. and Yang, J. (2004). Interaction Spaces in Data and Information Visualization. In Deussen, O., Hansen, C., Keim, D., and Saupe, D., editors, *Eurographics / IEEE VGTC Symposium on Visualization*. The Eurographics Association.

Wolfe, J. M., Kluender, K. R., Levi, D. M., Bartoshuk, L. M., Herz, R. S., Klatzy, R. L., and Lederman, S. J. (2009). *Sensation and Perception*. Sinauer Associates, 2nd edition.

Yi, J. S., ah Kang, Y., Stasko, J. T., and Jacko, J. A. (2007). Toward a deeper understanding of the role of interaction in information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1224–1231.

Zhang, Z., McDonnell, K. T., and Mueller, K. (2012). A network-based interface for the exploration of high-dimensional data spaces. In *IEEE Pacific Visualization Symposium*, pages 17–24. IEEE.

The Digital Aquarist: An Interactive Ecology Simulator

Julian Schikarski, Oliver Meisch, Sarah Edenhofer and Sebastian von Mammen

Organic Computing, University of Augsburg, Germany

Abstract

In this paper, we present an interactive simulation of the ecological cycle in a fish tank. Like the owner of a real fish tank, the user of the simulation has to balance several vital parameters of the aquatic system. Next to people interested in the world of aquatics in general, the simulation especially targets teenagers and aims at increasing their interest in ecosystems, and to contributing to their understanding of basic ecological principles. We engage the user introducing various gamification elements, including game-like UI elements, high scores, and a diligently adjusted reward system that allows for adding new inhabitants to the aquarium. Based on a user study, we evaluate our concept and layout possible improvements and extensions.

Introduction

A profound understanding of complex relationships and processes in ecological systems is an important factor for making informed, sustainable decisions. Like other empirical sciences, ecological research heavily relies on digital tools, including those for storage/retrieval, modelling and analysis (Jones et al. (2006)). Educators have been assembling a similar repertoire of digital tools for teaching ecological systems, whereas computational simulations are especially well suited to convey their inherent, often fragile complexity (Stevenson et al. (2014)).

In this paper, we present *The Digital Aquarist*, an interactive simulation of the ecological cycle in a fish tank. Following the concept of gamification, see e.g. Deterding et al. (2011); Groh (2012), we engage the user in the simulation by providing easy and rewarding access to the model context, to the model mechanics and especially to the offered user interactions. Along the same lines, we reduce the amount of prior knowledge required for a rewarding simulation experience to a bare minimum: (1) The user can easily explore the interdependencies between different organisms by himself. (2) The simulation is staged in a moderately sized fish tank that is often found in private homes (about 6.5% of households keep fish in the U.S. (AVMA, US (2012))).

We further distilled a model complex enough to convey foundational ecological relationships and the emergent system dynamics, yet simple enough to work for an introductory educational setting. The interaction possibilities are part of this model simplification: The user is encouraged to balance the different system variables by adding or removing organisms from the fish tank. Each animal or plant has a certain impact on the ecosystem by either reducing or increasing systemic parameter values, e.g. through breathing. The goal is to keep the schools of fish healthy over a long period of time while increasing the number of inhabitants, and thus the heterogeneity and the complexity of the ecosystems' population.

The remainder of this paper is structured as follows: In the next section, we discuss related work that influenced this project. Afterward, we present our modelling approach, the use of gamification elements, the realisation of aesthetic visualisation and the degree of complexity of the application. Based on our approach, we summarise our accomplishments. Finally, we outline how the simulation could be extended in the future.

Related Work

Creating a closed ecological cycle has been attempted by scientists in various projects. Biosphere 2 is a notable representative project of this kind (Allen and Nelson (1999)). It is an architectural and technological large-scale compound for exploring the interplay between human life and its environment in a closed ecological system. Biosphere 2 had originally been planned as a self-sustaining system. It is used as a research laboratory now, after two attempts to make it work have failed. The same idea but, at least commercially, more successful is the Ecosphere, an aquarium whose inhabitants are completely sealed off from any metabolic exchange with the environment. Only the sun light enters from the outside world and drives growth and transformation processes of the contained plants and animals (Schilthuizen (2009)). Although it is being disputed whether the life stock, the shrimp Halocardina Rubra, is surviving rather than slowly starving to death (Bailey-Brock and Brock (1993)).

Despite their minimalistic approaches and their emphasis on the exploration of well-defined ecological processes, neither Biosphere 2 nor the Ecosphere are apt for learning about and exploring ecosystem dynamics. One reason is the inaccessibility of the given systems. It is barely possible to change any of their compositions. Time scales are another reason: It takes long periods of time for ecological systems to stabilize, rendering it (even more) impractical to proactively change their settings and to explore their dynamics. For these reasons, providing interactive simulations of isolated problem domains, or *microworlds*, has become an important methodological approach in education and education research (Miller et al. (1999); Druin et al. (2014)).

While it has been emphasised that idealised model representations (as opposed to concrete ones) foster the development of generalisable insights (Goldstone and Son (2005)), *The Digital Aquarist* prioritises relatable, engaging aesthetic animation over the abstract display of an expectedly vivid, visually attractive ecosystem. Therefore, different from a preceding, NetLogo-based 2D aquarium model (Tan and Biswas (2007)), *The Digital Aquarist* models the aquarium and its inhabitants in an animated three-dimensional world.

Interactive parameter adjustment of a simulated aquarium has previously been used to study human learning and planning capacity (Vollmeyer et al. (1996)). It could show that promoting the free exploration of a dynamic system allows the user to gain general knowledge, whereas addressing specific tasks would solely foster specific knowledge. We harness this insight by allowing the users to freely explore our simulation and to only provide implicit stimuli to maintain the aquarium over time and with growing heterogeneity. However, as the investigation of intellectual capacities is not the *The Digital Aquarist*'s goal, we also reveal detailed information about the inhabitants of the simulation and their relationships, if inquired by the user.

Methodology

For the sake of accessibility, we focus on simple visuals and avoid overburdening the user with information which would, very likely, jeopardise the attractiveness of the simulation (Steele and Iliinsky (2010)). Instead of promoting formal analytical skills, we make sure to provide visible feedback similar to real-world experiences, including rampant algae growth upon eutrophication or starving fish. To keep the user both interested and involved, we built the simulation on the three “pillars of fun” (Koster (2013))—relatedness, competence, and autonomy: (1) Relatedness is established by the fact that aquaria may exist in households similar to the ones the potential users of the simulation call their home (see Figure 1). The great number of aquaria worldwide renders it likely that the users are even familiar with the concept of keeping ornamental fish, possibly also the notion that the aquarist needs to ensure an ecological balance. Finally, we establish a connection between the user and the simulation by showing that initially the virtual aquarium is empty and thus that he is responsible for each and every one of its inhabitants. (2) To promote the competence of the user, he needs to be challenged without giving rise to frustration. This goal is supported by the facts that *The Digital Aquarist* builds an ecosystem one step at a time, that the user always has the power to change its configuration back to a previous, simpler state, and that he can pro-actively inquire information about the aquarium's inhabitants and their relationships. To ease the user into the simulation scenario, he may enter a tutorial level from the main screen (Figure 2) and step through a guided tour shedding light on the impact of different species on the ecosystem. (3) *The Digital Aquarist* provides an inherently autonomous user experience in that it does not enforce the fulfilment of specific tasks but it lets the user explore the aquarium dynamics on his own. A high score system is provided that rewards the user's achievements but it does not limit the potential of exploration.

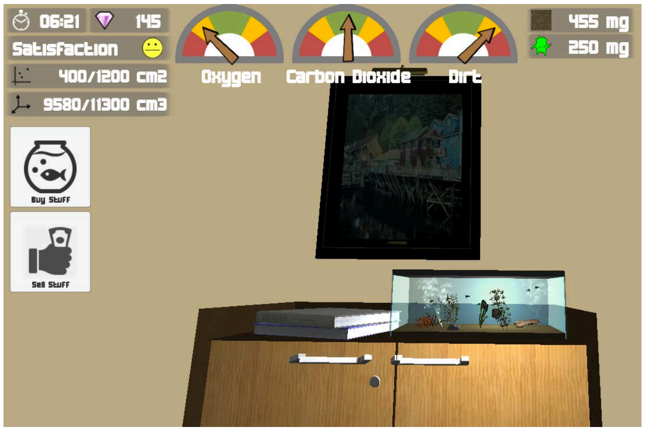


Figure 1: The simulated aquarium placed on a cupboard signals an everyday real-life scenario. The user interface aligned at the border of the view invites the user to join a playful simulation session.

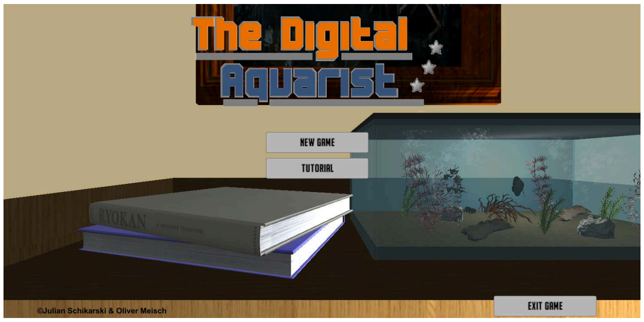


Figure 2: From the main menu of *The Digital Aquarist*, the user may access the high score list, enter a tutorial or join an endless explorative simulation session.

The Aquarium Model

In order to ensure an ecological equilibrium, several variables that describe a fish tank's state have to be maintained at certain levels. The main parameters are the levels of oxygen, carbon dioxide, nutrient matter in the seabed, the water volume, and the unoccupied space in the tank. Other important factors are the hardness of the water, its temperature, and the tank's light exposure. To allow the user to focus on key aspects of the ecological cycle, the latter two aspects are neglected in our model. In a real aquarium, these factors have to be adjusted by means of external devices.

The user can balance the aquarium parameters by adding and removing various animals and plants which all have their own way of interacting with the system. The amount of plants impacts the amount of nutrient matter in the water and the seabed, the levels of oxygen and carbon dioxide in the water, as well as the amount of unoccupied space in the tank. Seaweed breathes in carbon dioxide and breathes out oxygen, takes nutrient matter out of the seabed and releases small particles of nutrient matter into the water.

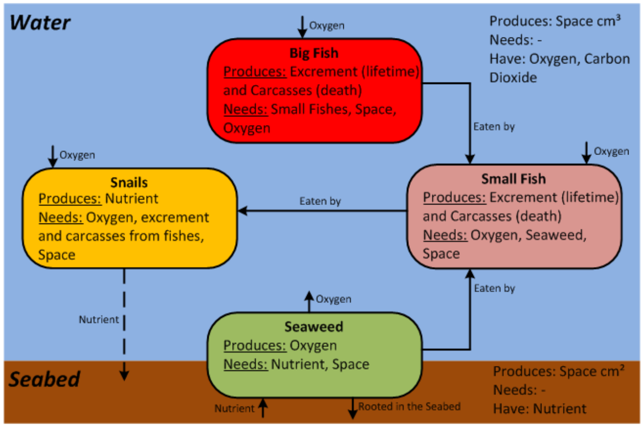

Water
Oxygen
Big Fish
Produces: Excrement (lifetime) and Carcasses (death)
Needs: Small Fishes, Space, Oxygen
Produces: Space cm³
Needs: -
Have: Oxygen, Carbon Dioxide
Eaten by
Oxygen
Oxygen
Snails
Produces: Nutrient
Needs: Oxygen, excrement and carcasses from fishes, Space
Eaten by
Small Fish
Produces: Excrement (lifetime) and Carcasses (death)
Needs: Oxygen, Seaweed, Space
Eaten by
Oxygen
Nutrient
Seaweed
Produces: Oxygen
Needs: Nutrient, Space
Seabed
Produces: Space cm²
Needs: -
Have: Nutrient
Nutrient
Rooted in the Seabed

Figure 3: Overview of our fish tank ecosystem model.

Figure 3 provides an overview of the inhabitants of the aquarium and their impact on the ecosystem. Nutrient matter in the water is consumed by the fish. The snails add the fish' excrements to the seabed. The seabed in turn serves as a nutritional basis for the seaweed. Small parts of the seaweed that break away and enrich the water are picked up by the fish again. The seaweed also produces the oxygen snails and fish breathe and absorbs the carbon dioxide they produce. This cycle can be disrupted by fish either eating smaller peers or seaweed in great quantities, which happens if there is not enough nutrient matter in the water.

Organismal Interdependencies The food intake of the fish scales with their size/age. Next to fish and plants, snails populate the virtual aquarium. Both micro-organisms and snails transform the fish' excrements in the water in nutrient matter that agglomerates in the seabed. This mechanism completes the nutrition cycle in the system. In order to provide ample visibility, the presence of snails also represents the transformative power of micro-organisms in our simulation. This means that snails are the only organisms accessible to the user that filter dirty water (from fish excrements) and feed nutrients into the seabed. In reality these processes would be addressed by both snails and micro-organisms. Same as fish, snails breathe in oxygen and breathe out carbon dioxide.

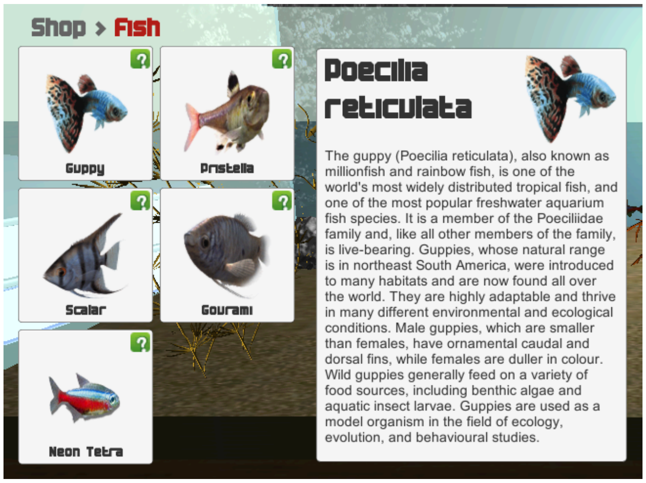
For the user to create a closed ecosystem, he needs to add members of each class of organisms and ensure that their mutual impacts keep a nice balance. Fish and snails need to breathe in a certain amount of oxygen and need the carbon dioxide to be below a certain level to keep from suffocating. Plants need to breathe in a certain amount of carbon dioxide, otherwise they suffocate. Fish need to absorb nutrient matter from the water, snails filter the fish' excrements, and seaweed absorbs nutrient matter from the seabed.

Additionally to the aforementioned interdependencies, each organism takes up a certain amount of space. When the fish tank gets too crowded, the fish will get stressed and will be unable to procreate. The procreation of fish adds another layer of complexity to the system. Due to the procreation of the fish, the user cannot easily anticipate the needed amount of inhabitants of the aquarium before starting the simulation. Therefore, the user needs to react to changing conditions on the fly, either by removing individual fish or by providing more nutrient matter, as well as snails and plants to find a new equilibrium.

Model information about the individual organisms is made available to the user on demand. A shopping interface allows to choose and add organisms to the ecosystem. The user needs to earn virtual currency to buy the organisms in the store. He earns coins by keeping animals and plants alive for as long as possible. This positive feedback mechanism ensures that the user is not immediately overwhelmed by a great number of organisms in the tank and also that fewer expensive organisms are added at first that are harder to cope with. At the same time, keeping a healthy ecosystem is directly translated into a rewarding sensation with actual impact on the interaction possibilities.

Currently, a selection of five fish is offered in the virtual store. Their impact on the ecosystem only differs due to their different sizes which in turn affects their metabolic rates. Otherwise, they all play the same role in the system. That means that all fish breathe in oxygen, consume nutrient matter, emit carbon dioxide, and leave excrements behind. Yet, the respective amounts vary from species to species. The store interface also provides additional information about the organisms, as exemplarily shown in Figure 4.

Modelling Metabolisms There are several model assumptions that have been made to keep the model complexity manageable. In particular, we assume a constant water tem-



The guppy (Poecilia reticulata), also known as millionfish and rainbow fish, is one of the world's most widely distributed tropical fish, and one of the most popular freshwater aquarium fish species. It is a member of the Poeciliidae family and, like all other members of the family, is live-bearing. Guppies, whose natural range is in northeast South America, were introduced to many habitats and are now found all over the world. They are highly adaptable and thrive in many different environmental and ecological conditions. Male guppies, which are smaller than females, have ornamental caudal and dorsal fins, while females are duller in colour. Wild guppies generally feed on a variety of food sources, including benthic algae and aquatic insect larvae. Guppies are used as a model organism in the field of ecology, evolution, and behavioural studies.

Figure 4: Additional information about a guppy fish is offered to the user in the virtual shopping interface.

perature of $20°C$, we do not consider the day/night cycle, we consider $9mg/l$ of oxygen as fully saturated freshwater (Dean L. Shumway (1964)), $50mg$ O_2 consumption per $100g$ of fish body weight per hour (Brett (1972)), $5mg$ O_2 production by $1g$ of algae per hour, and a $10sec$ integration step size of the simulation.

Table 1 lists the model variables involved in the calculation of the degree of oxygen saturation in the tank. The binary function $oCO_2(t)$ indicates whether or not a surfeit of carbon dioxide can be determined at time step t, i.e. a CO_2 value greater than or equal to twice the standard level of CO_2 is detected.

$h, w, d \in \mathbb{R}^*_+$	fish tank dimensions (cm)
$c \in \mathbb{R}^*_+ = h \times w \times d$	fish tank capacity (cm^3)
$li \in \mathbb{R}^*_+ = c/1000$	fish tank capacity (litres)
P	set of all plants
A	set of all animals
$O_2(t) \in \mathbb{R}^*_+ = li * 9$	amount of O_2 (in mg) at time $t = 0$
$i = 10$	integration step size (seconds)
$oCO_2(t) \in \{0, 1\}$	strong over saturation of CO_2 at time t

Table 1: Model variables for calculating oxygen saturation.

Based on the given variables, we calculate the amount of O_2 (in mg) at time t according to Equation 1. The oxygen saturation level is the ratio of current oxygen in the system relative to the initial oxygen level at $t = 0$. In conclusion, the oxygen saturation is influenced by the amount of oxygen produced and used by each organism in the fish tank. Exemplary values for the O_2 consumption of model organisms are listed in Table 2, whereas the intake is negative for algae since they produce oxygen.

$$\begin{aligned} O_2(t) = O_2(t-1) + \sum_{p \in P} O_2(p) - \sum_{a \in A} O_2(a) \\ - (\sum_{a \in A} O_2(a) * 0.25) * oCO_2(t-1) \end{aligned} \tag{1}$$

species	body weight	O_2 intake
Pterophyllum scalare	$300g$	$0.42\ mg/i$
Poecilia reticulata	$10g$	$0.014\ mg/i$
Aponogeton ulvaceus	$50g$	$-1.39\ mg/i$
Alternanthera reineckii	$100g$	$-2.78\ mg/i$

Table 2: Exemplary values of O_2 intake of model organisms.

Figure 5 shows an exemplary evolution of the oxygen saturation level. Until $t = 4$, 24 fish (12 *Pterophyllum scalare* and 12 *Poecilia reticulata*) slowly decrease the level of oxygen in the aquarium, despite the presence of two plants (*Aponogeton ulvaceus* and *Alternanthera reineckii*). At $t = 4$, one of the two plants dies off and the oxygen depletes twice as fast as before. The lack of oxygen leads to suffocation of 21 fish at $t = 7$ which results in the recovery of the oxygen saturation rate based on one remaining plant.

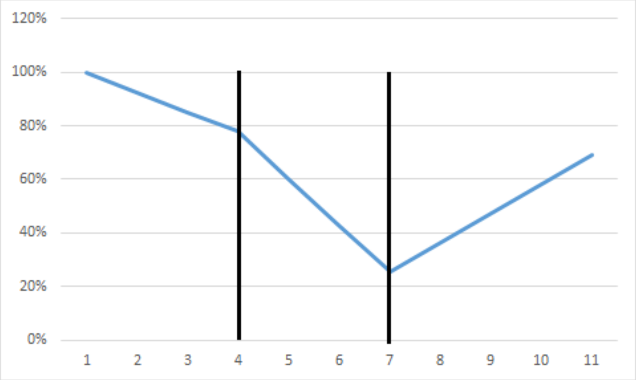


Figure 5: Oxygen saturation over time.

The amount of nutrients, the amount of dirt and the amount of carbon dioxide are computed in the same way as oxygen. Yet, the according equations consider the different organisms' impact on these variables. In case of nutrients in the water, the organisms take on the same role as for the oxygen level—plants increase their level, animals reduce it. Only the actual values of nutrient matter provided and consumed differ. The roles of the organisms are switched in terms of carbon dioxide, i.e. animals exhale CO_2 output and plants consume it. Dirt arises from the set of all fish

F and is diminished by the set of all snails S, resulting in Equation 2.

$$Dirt(t) = Dirt(t-1) + \sum_{f \in F} Dirt(f) - \sum_{s \in S} Dirt(s) \quad (2)$$

Simplifications Balancing complexity and accessibility, we have setup an approximative model. Therefore, we have to investigate the impact on the model accuracy conveyed to the user. The fish offered to the user to populate the aquarium resemble two popular ornamental species, the guppy and the scalare. The guppy mainly feeds on zooplankton which is living plankton, in the simulation it feeds on plankton produced by seaweed (fishbase.org (2015a)). Scalare are known to eat small fish as portrayed in the simulation (fishbase.org (2015b)). Snails do have a cleaning effect on the fish tank, yet they usually eat leftover food and algae (planetinverts.com (2015)). As mentioned before, the snails also visually represent the role of micro-organisms in the ecosystem, to empower the user to easily trace and influence the delicate dependency network.

Despite its simplifications, the current model conveys foundational interdependencies an aquarist needs to be aware of. Therefore, we feel the overarching goal of educating about ecological systems' dynamics is not weakened. Yet, we would like to identify and integrate new ways of high-level visualisations for improving the model accuracy without jeopardising *The Digital Aquarist*'s accessibility.

User Challenges

The learnings of *The Digital Aquarist* result from freely exploring, learning (primarily) by trial-and-error, and mastering a potentially great complexity of an ecosystem. They include a notion of the basic interdependencies of the interacting species as well as their evolution over time: Depending on the metabolic status of the aquarium and the configuration of its population, the effect of adding individual organisms to or removing them from the tank is delayed. In order to successfully manage the aquarium, the user has to anticipate these developments. This is especially challenging as each organism influences more than one system variable.

Without user intervention, the collapse of the ecosystem might accelerate, for example by hungry guppies eating seaweed as seen in Figure 6. Devouring seaweed further lowers the amount of plankton in the water, thereby making food an even scarcer resource. This example illustrates how easy it is to disturb an ecologically balanced system and that restoring that balance is not easy, especially if many different species are involved.

User Interface

As seen in Figure 1, three system variables (O_2, CO_2, and dirt) are represented by gauges which enable the user to check the current levels at a glance. Coloured segments indicate the criticality of the respective variables, whereas green indicates a favourable situation, yellow requires the user's attention and red underlines a fatal system state. An increasing level of dirt is also reflected by the water gradually turning green (Figure 7). The icons on the left-hand side of the screen indicate the duration of the simulation in progress, the cumulative score and the overall satisfaction of all organisms in the tank. These information are represented by the timer, the diamond and the smiley icon, respectively. The levels of nutrient matter in the seabed and water are displayed as numbers on the right-hand side of the screen, as the only restriction for them is not to reach zero. Since the fish tank provides limited space it is important for the user to know how many more fish and plants he can add to the system. Therefore on the lefthand side there are indicators how much space in cm^2 is left for plants in the seabed and how much space in cm^3 is left in the water for fish. The user interface enables an intuitive understanding of the status quo and quickly provides feedback about the ecosystem's evolution.

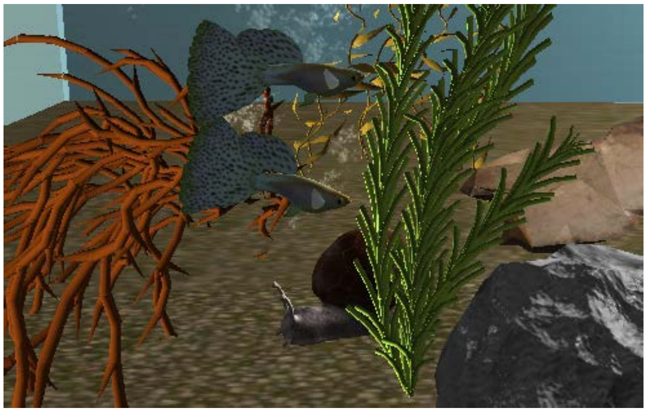


Figure 6: Guppies eating seaweed due to a lack of plankton in the water.

It is important to invest effort into the visual appeal of an interactive aquarium simulation—after all, ornamental fish are not only kept for the fascination for living organisms only but also for their elegance and beauty. Figure 8 shows the flocking of fish which mimics a life-like behaviour and a realistic look of aquarium. The flocking behaviour of fish was implemented according to the boid concept by Reynolds (1987). Here, each individual moves in accordance with its neighbours (Figure 9). In particular, it is urged to keep a minimum distance from its peers (*separation urge*), to flock towards the average location of its neighbours (*cohesion urge*) and to align its velocity with their average velocity (*alignment urge*). We generate circular waypoints throughout the aquarium to let the schools' movement appear naturally.

Soothing background music creates an inviting, relaxing

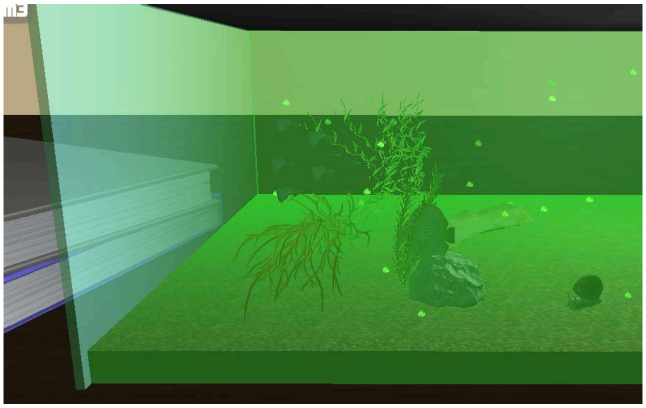


Figure 7: The water gradually turns green with an increasing degree of dirt.

atmosphere. Typical sounds of aquarium pumps and occasional oxygen bubbles popping on the surface help the user feel immersed into the simulation.

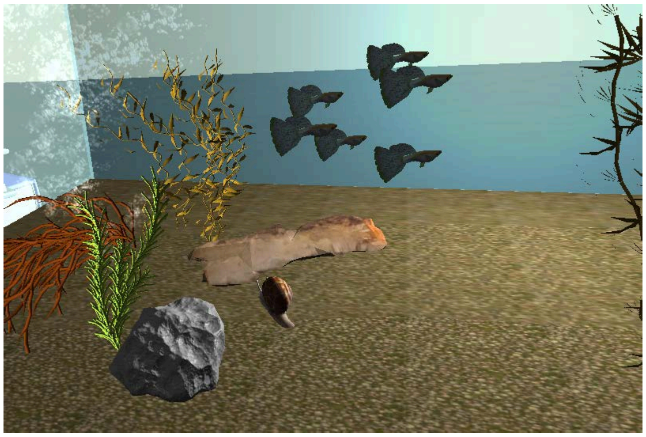


Figure 8: A school of guppies animated in accordance with the boids model (Reynolds (1987)).

Discussion

The user can effectively balance the system variables by adding and removing organisms to and from the aquarium. The game mechanics include possibilities to balance several parameters and observe their effects on the ecosystem of an aquarium. Let us have a look at the amount of oxygen. If a fish is added to the aquarium, the amount of oxygen decreases (dependent on the size of the fish). If the oxygen enters a critical sector—shown as the gauge in Figure 1 and explained in Section —the user can counteract by adding plants. The plants in turn increase the level of dirt (see Figure 7), what can be neutralised by increasing the number of snails, and so forth. Some simplifications, e.g., not considering the influence of light hitting the aquarium, help to limit the complexity to an understandable level.

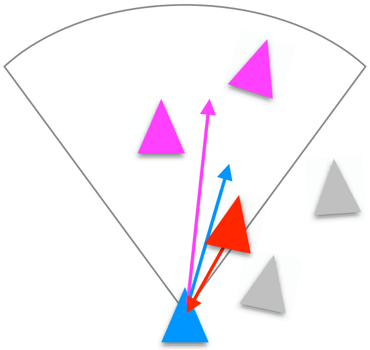


Figure 9: The boid flocking model considers cohesion towards perceived neighbours (pink arrow), separation from peers that are too close (red arrow) and alignment with the neighbours' average velocity (blue).

The user is rewarded for using more complex scenarios by a scoring/virtual currency system. In particular, higher scores are achieved when hosting bigger fish like scalares rather than relatively small guppies. The earned points can be spent on further additions to the aquatic ecology. We made *The Digital Aquarist* available online and invited colleagues and acquaintances by means of email lists and social media postings to evaluate it. In the according online survey, 31 testers provided anonymous feedback.

Table 3 shows their ratings regarding general aspects of *The Digital Aquarist*. From left to right, the percentages of testers reflect which aspects were "very poor, poor, fair, good, or excellent" (represented as "$--, -, o, +, ++$" in the table). Taking ratings as "good" or "excellent" into consideration, a majority felt that the topic of the game was a good choice, the model complexity was appropriate, and *The Digital Aquarist* was easy to use, 41.39% said it provided some fun. The aesthetics of the game and the learning effect were mainly rated as "fair" or "good" (74.19%, respectively 67.75%), the intuitiveness of the game mechanics was rated as "poor" by 35%, "fair" by 19.35% and "good" by 25.81%. The latter fact stroke us as particularly interesting as the game mechanics are aligned with the model facts the users would learn—if they were considered intuitive in the first place, there would be little knowledge that could be learned. And indeed, many testers felt that they had learned about ecological balance ($44, 44\%$) and about aquarium ecologies in particular ($48, 15\%$). These opinions were supported by some multiple choice questions that inquired about the aquatic organisms' interactions. A great majority of testers recognised facts about the metabolism of fish, snails, and seaweed. Yet, their feeding habits were not as clearly understood. For instance, 11% of the testers erroneously thought snails contributed to the pollution of the water, only about 40% realised that fish eat other fish (which only happens if other food sources become scarce), and only 26% recognised that seaweed was involved in nutrient production.

	−−	−	o	+	++
Game Topic	0	12,9	25,81	**45,16**	16,13
Aesthetics	3,23	6,45	**38,71**	35,48	16,13
Model Complexity	0	12,9	32,26	**48,39**	6,45
Fun	16,13	16,13	25,81	**29,03**	12,9
Learning Effect	12,9	12,9	**41,94**	25,81	6,45
Intuitiveness of Game Mechanics	3,23	**35,48**	19,35	25,81	16,13
Ease of Use	6,67	10	20	**40**	23,33

Table 3: Ratings (in %) of different aspects of *The Digital Aquarist* provided by 31 anonymous testers.

Summary & Future Work

The Digital Aquarist provides a small-scale ecosystem based on simplified metabolic models. An accessible user interface is supported by animation, visualisation and audio tracks to provide for an open-ended simulation experience that conveys the delicate balance needed to maintain a complex system.

A user survey of our first implementation of *The Digital Aquarist* indicates that we have successfully addressed certain challenges, including finding a proper level of abstraction of the simulated model as well as providing the necessary accessibility. Yet, we also appreciate that there is leeway for further improvement. Component-based development environments render it quite easy to setup intricate tracking shots that could allow the user to follow individual fish or snails, to experience the ecological processes in a more immersive manner and to reveal interactions close-up. These perspectives could be supported by intricate animations, for instance based on particle systems, to clearly visualise organismal activities such as nibbling, eating, and excreting. In addition, diagrammatic augmentation could drastically speed up the learning process, indicating the relationships among the organisms on demand. In order to keep the user interested over a long period of time, the repertoire of available species, decorative items, and technical add-ons could grow after successfully mastering a balance for a given timespan. Along these lines, one should even consider providing different sizes, shapes and kinds of aquariums. The iconic fishbowl bears different possibilities and challenges than a saltwater tank.

Our simulation can be used by teachers and their biology classes in secondary schools to learn about ecological cycles. We have scheduled a demo/play event for teenagers, our targeted user group. Based on its success, we are planning the public, mobile release of *The Digital Aquarist*.

References

Allen, J. and Nelson, M. (1999). Overview and design biospherics and biosphere 2, mission one (1991–1993). *Ecological Engineering*, 13(1):15–29.

AVMA, US (2012). *Pet Ownership & Demographics Sourcebook*. American Veterinary Medical Association, Schaumberg, IL.

Bailey-Brock, J. H. and Brock, R. E. (1993). Feeding, reproduction, and sense organs of the hawaiian anchialine shrimp halocaridina rubra (atyidae).

Brett, J. (1972). The metabolic demand for oxygen in fish, particularly salmonids, and a comparison with other vertebrates. *Respiration Physiology, Volume 14, Issues 1-2, Pages 151-170.*

Dean L. Shumway, Charles E. Warren, P. D. (1964). Influence of oxygen concentration and water movement on the growth of steelhead trout and coho salmon embryos. *Transactions of the American Fisheries Society, Volume 93, Issue 4, pages 342-356.*

Deterding, S., Sicart, M., Nacke, L., O'Hara, K., and Dixon, D. (2011). Gamification. using game-design elements in non-gaming contexts. In *PART 2———Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, pages 2425–2428. ACM.

Druin, A., Blikstein, P., Fleer, M., Read, J. C., Thomsen, B. S., Johnson, B. D., and Resnick, M. (2014). How can interaction with digital creative tools support child development?:(closing panel). In *Proceedings of the 2014 conference on Interaction design and children*, pages 361–361. ACM.

fishbase.org (2015a). Guppies. `http://www.fishbase.org/Summary/SpeciesSummary.php?ID=3228&AT=guppy`.

fishbase.org (2015b). Scalare. `http://www.fishbase.org/Summary/speciesSummary.php?ID=4717&AT=scalare`.

Goldstone, R. L. and Son, J. Y. (2005). The transfer of scientific principles using concrete and idealized simulations. *The Journal of the Learning Sciences*, 14(1):69–110.

Groh, F. (2012). Gamification: State of the art definition and utilization. *Institute of Media Informatics Ulm University*, pages 39–47.

Jones, M. B., Schildhauer, M. P., Reichman, O., and Bowers, S. (2006). The new bioinformatics: integrating ecological data from the gene to the biosphere. *Annual Review of Ecology, Evolution, and Systematics*, pages 519–544.

Koster, R. (2013). *Theory of fun for game design*. O'Reilly Media, Inc.

Miller, C. S., Lehman, J. F., and Koedinger, K. R. (1999). Goals and learning in microworlds. *Cognitive Science*, 23(3):305–336.

planetinverts.com (2015). Horned nerite snail. `http://www.planetinverts.com/horned_nerite_snail.html`.

Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics*, 21(4):25–34.

Schilthuizen, M. (2009). Life in little worlds. *The Loom of Life: Unravelling Ecosystems*, pages 1–10.

Steele, J. and Iliinsky, N. (2010). *Beautiful visualization*. O'Reilly Media, Inc.

Stevenson, R. D., Klemow, K. M., and Gross, L. J. (2014). Harnessing bits and bytes to transform ecology education. *Frontiers in Ecology and the Environment*, 12(5):306–307.

Tan, J. and Biswas, G. (2007). Simulation-based game learning environments: Building and sustaining a fish tank. In *Digital Game and Intelligent Toy Enhanced Learning, 2007. DIGITEL'07. The First IEEE International Workshop on*, pages 73–80. IEEE.

Vollmeyer, R., Burns, B. D., and Holyoak, K. J. (1996). The impact of goal specificity on strategy use and the acquisition of problem structure. *Cognitive Science*, 20(1):75–100.

Toward a behavior-based approach to the origins of life and the genetic system

Tom Froese

Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas
Universidad Nacional Autónoma de México
t.froese@gmail.com

Abstract

In the origin of life community there has been a dispute about whether metabolism or replication came first. Yet both of these approaches are in implicit agreement that the first forms of life were basically passive. That shared assumption has begun to be challenged by a new generation of metabolism-first approaches, emphasizing that movement and adaptive behavior could have played an important role right from the start. After introducing recent research on this behavior-based approach to the origin of life, I offer a preliminary assessment of what this new approach implies for the origins of the genetic system.

Current state of the field

Metabolism- and replicator-first approaches, while differing in many respects, implicitly agree that the first forms of life were passive and encapsulated individuals. Both failed to consider the intermediate timescales between chemical self-constitution and population evolution: no mention is made of motility nor of development (Froese et al., 2011). Yet new metabolism-first approaches emphasize that motility and adaptive behavior could have played a crucial role from the start (Froese et al., 2014; Hanczyc, 2011; Egbert et al., 2012). Such a *behavior-based* approach can refresh thinking on several classic issues.

For example, it was long believed that pre-biotic evolution of a hypercycle of autocatalytic processes is unstable because it can be invaded by parasitic compounds. But later on it was realized that this is only the case in a well-mixed (non-spatial) environment, while spatial embedding can make hypercycles immune to parasites. We can push this change in perspective even further. It has been shown that it is possible to make use of the instabilities introduced by the parasites as a source of spontaneous motility, a capacity which can be adaptive under some conditions (e.g., Froese et al., 2012). Indeed, there may even have been forms of minimal cognition already at the origin of life (Hanczyc & Ikegami, 2010).

The next challenges

From the behavior-based perspective it appears likely that the origins of the genetic system were related with the origins of adaptive control of behavior during an organism's lifetime. In particular, current examples of behavior-based proto-life have a very limited set of states. Accordingly, they cannot respond differentially based on their long history of interactions. What is required is a primitive kind of memory system that permits state-based adaptive modulation of sensorimotor interactions. Could such a memory system be partially heritable and have provided the foundation for the evolution of a genetic system?

This hypothesis is consistent with the proposal that genetic evolution first began with the composite chemical phenotype serving as its own holistic genotype (Segré et al., 2000). The next milestone of this composite proto-metabolism would be the evolution of state-based adaptive behavior which enhances its survival, with some parts beginning to turn into a dedicated memory system. I speculate that aspects of this system might have eventually evolved into a relatively fixed genotype with gene sequences that 'code' for specific properties. To see how this could have happened in principle, we can draw inspiration from computer models of the origins of another compositional symbol system, language (Christiansen & Dale, 2004). Here too there is a shift from a gene-centric bias to more interactive approaches, e.g. the iterated learning model. This simplified process of cultural evolution can give rise to a syntactical communication system that 'codes' for specific features when starting from the exchange of arbitrary string-feature pairings. It still remains to be seen if such an interactive self-organizing principle could have applied to the iterative self-replication or horizontal gene transfer performed by the first proto-cells.

References

Christiansen, M. H., & Dale, R. (2004). The role of learning and development in language evolution: A connectionist perspective. In D. K. Oller & U. Griebel (Eds.), *Evolution of Communication Systems* (pp. 91-110). Cambridge, MA: MIT Press.

Egbert, M. D., Barandiaran, X., & Di Paolo, E. A. (2012). Behavioral metabolution: The adaptive and evolutionary potential of metabolism-based chemotaxis. *Artificial Life, 18*, 1-25.

Froese, T., Ikegami, T., & Virgo, N. (2012). The behavior-based hypercycle: From parasitic reaction to symbiotic behavior. In C. Adami et al. (Eds.), *Artificial Life 13* (pp. 457-464). MIT Press.

Froese, T., Virgo, N., & Ikegami, T. (2011). Life as a process of open-ended becoming: Analysis of a minimal model. In T. Lenaerts et al. (Eds.), *Advances in Artificial Life, ECAL 2011* (pp. 250-257). Cambridge, MA: The MIT Press.

Froese, T., Virgo, N., & Ikegami, T. (2014). Motility at the origin of life: Its characterization and a model. *Artificial Life, 20*(1), 55-76.

Hanczyc, M. M. (2011). Metabolism and motility in prebiotic structures. *Philosophical Transactions of the Royal Society B, 366*, 2885-2893.

Hanczyc, M. M., & Ikegami, T. (2010). Chemical basis for minimal cognition. *Artificial Life, 16*, 233-243.

Segré, D., Ben-Eli, D., & Lancet, D. (2000). Compositional genomes: Prebiotic information transfer in mutually catalytic non-covalent assemblies. *Proc. Natl. Acad. Sci. USA, 97*, 4112-4117.

An Efficient Ant Colony System for Edge Detection in Image Processing

Yara Khaluf[1] and Syam Gullipalli[2]

University of Paderborn, Paderborn, Germany
[1]yara.khaluf@uni-paderborn.de
[2]syam@mail.uni-paderborn.de

Abstract

Edge detection is a fundamental procedure in image processing, machine vision, and computer vision. Its application area ranges from astronomy to medicine in which isolating the objects of interest in the image is of a significant importance. However, performing edge detection is a non-trivial task for which a large number of techniques have been proposed to solve it. This paper investigates the use of Ant Colony Optimization — a prominent set of optimization heuristics — to solve the edge detection problem. We propose two modified versions of the algorithm Ant Colony System (ACS) for an efficient and a noise-free edge detection.

1 Introduction

Edge detection is a fundamental process in analyzing images. It attempts to find points at which the image brightness has discontinuities. These discontinuities allow changes in pixels intensities which may define the boundaries of an object. Applying edge detection may reduce significantly the amount of data to be processed by filtering out the information that is less relevant and preserving the important structural properties of an image. Therefore, edge detection is involved as an essential stage in a wide range of applications. Examples include medicine applications, pattern recognition, machine vision, image analysis, automotive applications, and others. Furthermore, edge detection should be performed in a reliable way as the validation and the efficient completion of the following stages in the image processing rely on it. At the same time, obtaining ideal edges from real life images with a moderate complexity is a challenging task.

In general, complex mathematical functions such as the first and the second order derivatives of the image are used for performing edge detection. Moreover, smoothing (filtering) functions are required to remove the noise obtained from the detection process. Applying such techniques increases the complexity of the detecting process and they may fail in maximizing the number of detected edges. Ant Colony Optimization Colorni et al. (1991) is a set of heuristics for optimization that has proven its efficiency in a large number of areas including scheduling problems Martens et al. (2007); Blum (2005), vehicle routing problem Toth and Vigo (2002); Secomandi (2000), assignment problems Stützle and Hoos (2000), and others. ACO is inspired by the foraging behavior of ants in which ants leave pheromone trails on the ground while searching for food in order to guid the search of other ants to the best food sources. There are several ACO algorithms presented in the literature, see Dorigo and Stützle (2004) and Christian (2005). They vary either in the way they select their next choice in the solution space or in their way of updating the pheromone trails. In this paper, we use the particular variant Ant Colony System (ACS) Dorigo and Gambardella (1997) to solve the problem of edge detection. ACS applies two rules for the pheromone update: local and global, in order to achieve a better search of the problem space. Additionally, its probabilistic decision rule focuses on both exploring the solution space and exploiting previous solutions. We propose two modified algorithms of ACS, which are developed to obtain an efficient edge detection with a minimized complexity.

The rest of the paper is organized as follows: section 2 reviews a list of related works in the field of edge detection. Section 3 illustrates the Ant Colony System used in solving the edge detection problem. In Section 4 we present the first algorithm (FACS) proposed to solve the edge detection based on ACS. Experimental results of FACS are reported in Section 5. The Extended FACS (the second algorithm proposed) is presented in Section 6 and its experimental results are reported in Section 7. A comparison between the performance of the algorithms is given in Section 8 and the paper is concluded in section 9.

2 Related Work

Several approaches have been introduced in the literature for edge detection and the solutions proposed were mainly based on complex mathematical techniques. Two of the wide-used methodologies are the gradient and the Laplacian. In the gradient, detecting edges is performed by searching for the maximum and minimum in the first derivative of the image. Whereas in the Laplacian, it is done by searching for the zero crossings in the second derivative of the image Ziou and Tabbone (1998). One of the prominent algorithms that has applied the first order derivative for edge detection is the Canny's algorithm Canny (1986). This algorithm has introduced an edge detection operator using the calculus of variations, Gelfand et al. (2000). Canny's algorithm applies a set of mathematical functions for detecting the edges and requires image filtering for removing the noise. Despite being one of the most applied edge detection techniques, Canny's algorithms is associated with a set of preconditions for enabling its application in addition to the high complexity associated with the execution of this algorithm. Prewitt has introduced in Prewitt (1970) another edge detection operator that uses the first order derivative of the image by computing an approximation of the gradient of the image intensity function. Another edge detection operation that computes an approximation of the gradient of an image through discrete differentiation is the one introduced by Roberts cross in Davis (1975). The main disadvantage of the first order derivative operators is their sensitivity to the noise while detecting both the edges and their orientations. On the other hand, detecting the image edges using the second order derivative of the image was first presented in Haralick (1984). This technique captures the rate of change in the intensity gradient and looks for the zero crossing of the second derivative of the image. The Marr-Hildreth operator Marr and Hildreth (1980) is a well-known operator that uses the second order derivative by convolving the image with the Laplacian of the Gaussian function. The Marr-Hildreth operator suffers from the possibility of generating responses that do not correspond to edges and are referred to as false edges. The second order derivative operators suffer, as the first order derivative ones, from their sensitivity to noise. Therefore using filtering techniques becomes a necessity for both types of operators, which increases the complexity of the edge detection process.

Different from the complex mathematical approaches used for the sake of edge detection, Ant Colony Optimization has offered efficient heuristics to deal with the problem. For example, the authors in Rezaee (2008) have investigated the use of a particular ACO variant, referred to as the ant system (AS) to detect edges. In Nezamabadi-pour et al. (2006) a modified ant colony system has been presented to solve the edge detection problem. The authors have derived an experimental relationship between the size of the image and the algorithm parameters. The quality of the detection was further improved in the ant colony algorithm presented in Tian et al. (2008). Another work in which ant colony system was involved in detecting edges was in Zhuang (2004), in which a perceptual graph was used to represent the processed image. A hybrid edge detection using Canny edge detector and an ant colony algorithm was presented in Manish et al. (2013). Even quantum computing was proposed to be used in combination with ant colony system to detect edges in Jian et al. (2012). In this work, the authors have applied matrix multiplications and complex mathematical functions such as trigonometric functions in order to detect edges. Many experiments have proven that Ant Colony System (ACS) — a variant of ant colony algorithms — dominates Ant system (AS) for edge detection problems, as we can see in Baterina and Oppus (2010). In this work, the authors investigated the ACS algorithm for solving edge detection problems. However, noise filtering was still a necessary step for obtaining a feasible result.

The work presented in this paper proposes the use of two modified versions of the ACS algorithm for solving edge detection problems. Moreover, it offers a comparison between the performance of the proposed algorithms and the performance of the ACS system presented in Baterina and Oppus (2010) (without noise filtering). A remarkable improvement, in terms of both quality and complexity, is achieved.

3 Ant Colony System (ACS)

Ant colony algorithms are population based meta-heuristics which share two main components: the probabilistic decision rule and the pheromone update rule. The probabilistic decision rule is the rule used by each ant to decide concerning its next step in the solution space. This decision is performed probabilistically based on both the problem heuristic information and the exploitation of the experiences of other ants that have explored the space before. Updating the pheromone trails includes two operations: pheromone evaporation to forget about the previous bad solutions and pheromone deposit to reinforce the good solutions. The pheromone trails, left by the ants are exploited by other ants to improve the quality of their search.

Ant Colony System (ACS) is a particular variant of the ant colony algorithms which has its own probabilistic decision and pheromone update rules. The decision rule applied in ACS exploits, with a particular probability, the search experience accumulated by other ants. Thus, the probability that the ant moves to position j is defined as in the following:

$$j = \begin{cases} \arg\max_{j \in N_i^k} (\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}) & \text{if } q \leq q_0 \quad \text{(Exploitation)} \\ J & \text{otherwise} \quad \text{(Exploration)} \end{cases} \tag{1}$$

where q is a random variable uniformly distributed in $[0, 1]$, q_0 $(0 \leq q_0 \leq 1)$ is an algorithm parameter, N_i^k is the set of unvisited neighbors, and J is defined using the probabilistic decision rule of the ant colony algorithms, given by:

$$p_{ij}^{(k)}(t) = \frac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k}[\tau_{il}(t)]^{\alpha}[\eta_{il}]^{\beta}} \tag{2}$$

where τ_{ij} is the pheromone value deposited on the edge (i,j). η_{ij} is the heuristic information assigned to the edge (i,j). α and β are two parameters that determine the relative influence of both the pheromone trails and the heuristic information.

ACS applies two rules for updating the pheromone trails: the global update and the local update. The global pheromone update is applied after each iteration by only one ant (the best-so-far), as in the following:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}, \quad \forall(i,j) \in T^{bs} \tag{3}$$

where ρ is the evaporation parameter and $\Delta\tau_{ij}^{bs}$ is the amount of pheromone deposited by the best-so-far ant.

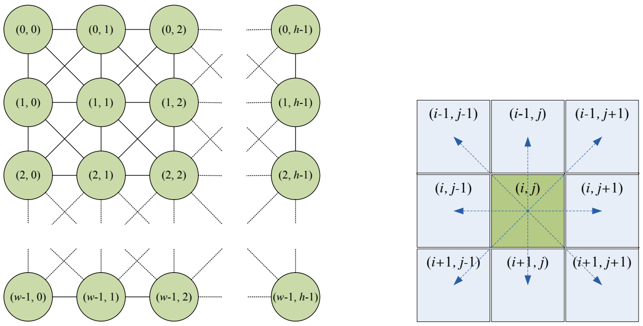
One of the main difference between ACS and other ant colony algorithms is the use of local pheromone. The purpose of using this pheromone is to reduce the probability for ants to select an edge that was selected before. The updating of local pheromone is done by each ant immediately after crossing a particular edge during the phase of the solution construction. The rule for updating the local pheromone is given by:

$$\tau_{ij} \leftarrow (1-\xi)\tau_{ij} + \xi\tau_0 \tag{4}$$

where ξ, $0 < \xi < 1$, and τ_0 are two parameters of the algorithm.

4 Focused ACS Algorithm for Edge Detection (FACS)

For enabling the application of any ant colony algorithm on edge detection problems, the image should be transformed into a graph. This graph is generated by introducing a matrix $I_{w \times h}$ that represents the intensity at each pixel of the image, as in Figure 1(a). Each pixel is considered as a node and is connected to its neighbors (horizontal, vertical, and diagonal) to form the graph edges.



***(a)** Graph representation of $w \times h$ image.*

***(b)** Feasible neighborhood of pixel (i, j).*

***Figure 1** – Graph representation of the edge detection problem.*

As illustrated in Figure 1(b), the neighborhood of each pixel is defined so that each ant can move in 8 directions if applicable. Repeated visits to the same node are restricted by maintaining a piece of memory by each ant. However, it is allowed in deadlock situations. In general, the heuristic information defined for most of the problems, which were solved with ant colony algorithms, is encoded on the edges of the graph that represents the problem. Differently, for edge detection problems, the heuristic information is encoded at the nodes of the graph (the pixels) and is defined based on the intensity value of the pixel, as in the following:

$$\eta_{ij} = \frac{V_c(I_{ij})}{V_{max}} \tag{5}$$

I_{ij} is the intensity of pixel $(i,\ j)$. V_{max} is the maximum intensity variation in the given image and $V_c(I_{ij})$ is the function of intensity variation around the pixel $(i,\ j)$ which is given by:

$$\begin{aligned} V_c(I_{(i,j)}) = &|I_{(i-1,j-1)} - I_{(i+1,j+1)}| + |I_{(i,j-1)} - I_{(i,j+1)}| \\ &+ |I_{(i+1,j-1)} - I_{(i-1,j+1)}| + |I_{(i+1,j)} - I_{(i-1,j)}| \end{aligned} \tag{6}$$

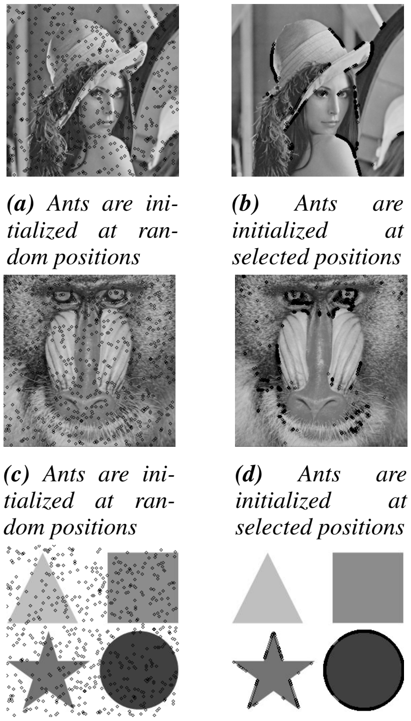
Consequently, the pixel with a higher intensity variation (its heuristic information value is higher) has a higher probability of being selected in the next step of the ant tour.

In ACS, the global pheromone update is applied only by the best-so-far ant. However for the edge detection problem, we modify this rule to allow all ants to update the pheromone trails at the end of each iteration. Since all the solutions found by ants represent potential edges in the image. Therefore, Equation Eq.(3) becomes:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \rho\Delta\tau_{ij}^{k} \tag{7}$$

If ant k has visited pixel $(i,\ j)$, then $\Delta\tau_{ij}^{k}$ is computed as the average of the heuristic values associated with pixel $(i,\ j)$. Otherwise, it is set to zero. Local pheromone update is applied by the ants after each move as defined in Eq.(4).

The ant colony algorithms presented in the literature for solving edge detection problem, such as in Baterina and Oppus (2010), require mostly a post-processing step of thresholding and filtering for removing the noise. Such techniques remove often edges apart from the noise and increases significantly the complexity of the detection process. In this paper, we propose an efficient algorithm (FACS) to overcome loosing the details of edges on resultant images and to reduce the complexity of the process. The core idea of FACS is to perform a focused distribution of the initial positions of ants on the image graph. The way the ants are places initially on the image graph affects significantly both the quality of the results and the complexity of the detection process. Most of the used ant colony algorithms apply a random distribution of the initial positions of ants. One of the main disadvantages of the random distribution is the noise emerges in the resultant images. Since the heuristic matrix of the image could be computed offline before starting the algorithm, FACS proposes to guide all ants to start at selected positions, at which the values of the heuristic information are at their maximum. FACS computes first the heuristic information matrix associated with the image graph. Afterwards, this heuristic information is ranked and the initial positions are selected based on the obtained ranking. This leads to concentrate the search by making it starts at positions related to the best solutions. Consequently, this will reduce the complexity of the search algorithm and increase the quality of the results. Figure 2 shows two standard test images (lena and mandril), in addition to one image of simple shapes that are drawn with an increasing intensity in order to clarify the applicability of the proposed algorithms. The Figure shows the initial positions of ants. In Figure 2(a), (c), and (e) the positions are generated randomly using a uniform distribution as done by most of the ant colony algorithms. Whereas, in Figure 2(b), (d), and (f) the positions are generated based on the ranked heuristic information, as described above.



(a) *Ants are initialized at random positions*

(b) *Ants are initialized at selected positions*

(c) *Ants are initialized at random positions*

(d) *Ants are initialized at selected positions*

(e) *Ants are initialized at random positions*

(f) *Ants are initialized at selected positions*

Figure 2 *– Initial positions of ants.*

The pseudocode of the FACS algorithm is shown in Algorithm 1, where initially K artificial ants are placed at particular positions that are selected based on the ranked heuristic information. The algorithm runs for N iterations and in each iteration each ant constructs its specific solution through choosing its steps probabilistically using Eq.(1) and Eq.(2). After that, the ant updates the local pheromone trails using Eq.(4). At the end of each iteration the pheromone trails are updated globally using the solutions found by all ants as in Eq.(7).

Algorithm 1: Pseudocode of FACS for edge detection in images.

```
Initialization at focused positions; { Compute the heuristic information
                                       Rank the heuristic information
                                       Select initial positions
forall the iterations $n$ in 1:$N$ do
    forall the construction steps $l$ in 1:$L$ do
        forall the ants $k$ in 1:$K$ do
            Choose and move to the next pixel;
            Update local pheromone;
    Update global pheromone values on visited pixels;
```

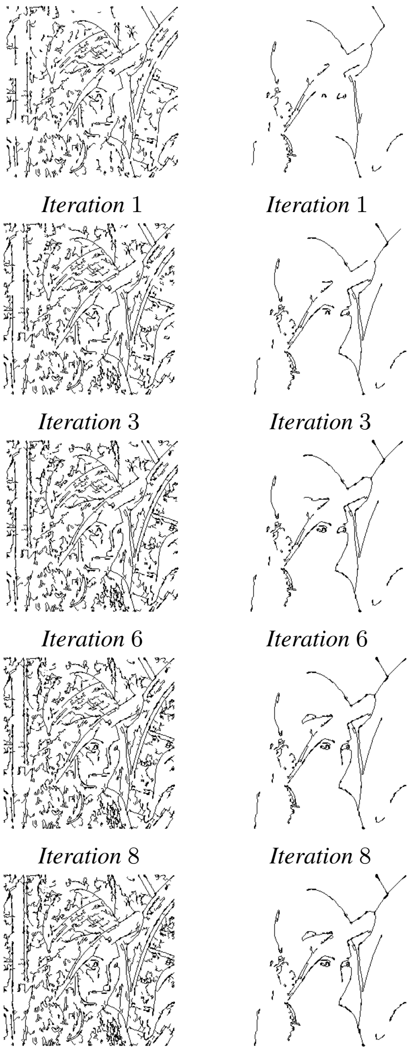
5 Experimental Results with FACS

We perform a set of experiments on the images shown in Figure 2 using two edge detection algorithms: the one presented in Baterina and Oppus (2010) and the FACS algorithm. The results obtained from applying the Baterina and Oppus (2010) algorithm are reported without performing any thresholding or filtering in order to compare them with the results obtained using FACS, which does not undergo any filtering process.

We have performed a parameter sensitivity analysis for FACS, in which we have varied the parameters over the following ranges: $q_0 \in \{0.1, 0.3, 0.5, 1\}$, $\alpha, \beta \in \{0.1, 0.3, 0.5, 0.7, 1, 2, 5\}$, $\tau_0 \in \{0.5, 1, 10, 100\}$, and $K \in \{128, 256, 512, 1024\}$. We have noticed that while changing the values of β or q_0 doesn't have a significant influence on the results, high values of α or τ_0 allow to preserve the edges found in pervious iterations. On the other hand, the time it takes to find good result is a tradeoff between the number of ants and the number of iterations. ρ and ξ were tested pairwise $(\rho, \xi) \in \{(0.01, 0.01), (0.1, 0.05), (1, 0.01), (1, 0.1), (1, 1)\}$ and experiments showed that better results were obtained for a high ρ and a relatively low ξ. For the implementation, we adopt the best parameter settings found in Baterina and Oppus (2010) and those are given in the following:

Initial pheromone value (τ_0) =	0.1
Number of ants (K) =	512
Number of iterations (N) =	10
Number of constructions (L) =	40
Parameter influencing pheromone trail (α) =	1
Parameter influencing heuristic information (β) =	1
Pheromone decay coefficient (ξ) =	0.05
Pheromone evaporation coefficient (ρ) =	0.1
Degree of exploration (q_0) =	0.7

We have $K = 512$ ants initialized on K positions that are selected in a decreasing order of the heuristic values. Those ants are initialized randomly in the algorithm presented in Baterina and Oppus (2010). Figures 3 and 4 show the binary images resulted over different iterations of both ACS and FACS algorithms, where neither thresholding nor filtering techniques were applied. The results show that all the edges found by ants in FACS are potential edges and there was a significant improvement in reducing the detection complexity and in the quality of the results (no noise) compared to the ACS algorithm presented in Baterina and Oppus (2010) (without filtering). After a particular number of iterations, the results from FACS may stop to improve (stagnation) as we can see in Figure 3 for the iterations 8 and 10.



Iteration 1 *Iteration* 1

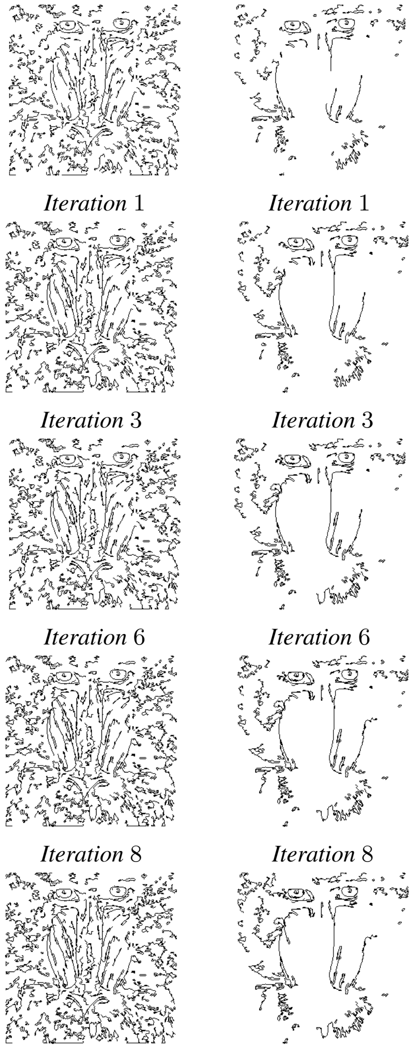
Iteration 3 *Iteration* 3

Iteration 6 *Iteration* 6

Iteration 8 *Iteration* 8

Iteration 10 *Iteration* 10

Figure 3 *– The binary images of the image "Lena" obtained over different iterations for both Baterina and Oppus (2010) algorithm and FACS.*



Iteration 1 *Iteration* 1

Iteration 3 *Iteration* 3

Iteration 6 *Iteration* 6

Iteration 8 *Iteration* 8

Iteration 10 *Iteration* 10

Figure 4 *– The binary images of the image "Mandril" obtained over different iterations for both Baterina and Oppus (2010) algorithm and FACS.*

6 The Extended FACS

Initializing ants at the best heuristic positions has shown a significant improvement over methods that initialize ants at random positions. However, after certain number of iterations the results may stagnate. The reason for this potential stagnation is associated with the reduced probability for the ants to detect edges in the next iterations, while the initial positions selected for those ants did not lead to detect edges in pervious iterations. In cases where the initial positions selected for the ants are adequate for detecting the edges in the image, FACS performs at its optimal. However when it is not the case, potential edges may not be found by the solutions generated over the different iterations.

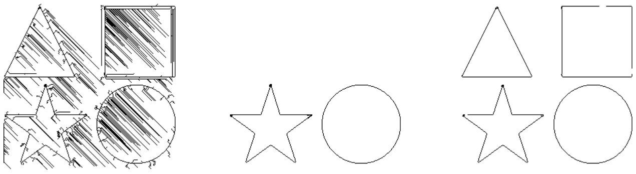
In order to solve this stagnation problem, we introduce the Extended FACS algorithm, in which the positions of the ants might be re-initialized after each iteration. The algorithm works as in the following: After the global pheromone is deposited at the end of the i-th iteration, we compute the average amount of pheromone deposited by all ants and which is denoted by $\tau_{avg}(i)$. The ants are ranked based on the quality of their solutions in the i-th iteration using the average $\tau_{avg}(i)$. Ants which have deposited a larger amount of pheromone than $\tau_{avg}(i)$ will start from the positions (pixels) they have finished at in the latest iteration. Whereas, ants which have deposited less pheromone than $\tau_{avg}(i)$ are place on new initial positions in the next iteration $i+1$. The new positions assigned to those ants are selected based on the heuristic information computed offline. Thus, They will be placed at the pixels with the next highest heuristic values. The memories of the ants which are placed at new positions are reset to allow them to re-visit the pixels they may have visited in the previous iterations. This extension allows the ants which have performed above the average to continue the search they have started and to detect connected edges. Whereas, the ants which have performed below the average, they have most likely detect no connected edges. Therefore, they should be placed at new positions for increasing their probability of finding new potential edges. The pseudocode of the Extended FACS algorithm is shown in Algorithm 2.

Algorithm 2: Pseudocode of the Extended FACS for edge detection in images.

```
Initialization at focused positions;  { Compute the heuristic information
                                      { Rank the heuristic information
                                      { Select initial positions
forall the iterations n in 1:N do
    forall the construction steps l in 1:L do
        forall the ants k in 1:K do
            Choose and move to the next pixel;
            Update local pheromone;
    Update global pheromone values on visited pixels;
    Assign low ranked ants    { Compute the average pheromone
    new initial positions;    { Rank the ants
                              { Re-assign initial positions
```

7 Experimental Results with the Extended FACS

In this section, we apply the Extended FACS to validate the algorithm performance in terms of the solution quality and the ability to deal with the stagnation problem. We adopt the same parameter settings as in Section 5. First, Figure 5 shows the results of applying all of ACS, FACS and the extended FACS on the shapes image. It depicts the results obtained after several iterations of each of the algorithms. In Figure 5(a), we can notice the noise generated when using ASC without filtering. On the contrary, we obtain free-of-noise binary images, when using any of FACS or the extended FACS algorithms, as we can see in Figures 5(b) and (c). Because of initializing the ants at the same positions with the highest heuristics (highest intensity), the ants used by FACS were able to detect edges only on the circle and the star shapes. This is not the case when applying the extended FACS, in which ants are re-intialized at new positions when their previous positions were not promising, see Figure 5(c). Additionally, having a larger number of ants will lead to decrease the number of iterations required by both FACS and the extended FACS to obtained the optimal results and to allow FACS to detect more edges (on the square and the triangle).



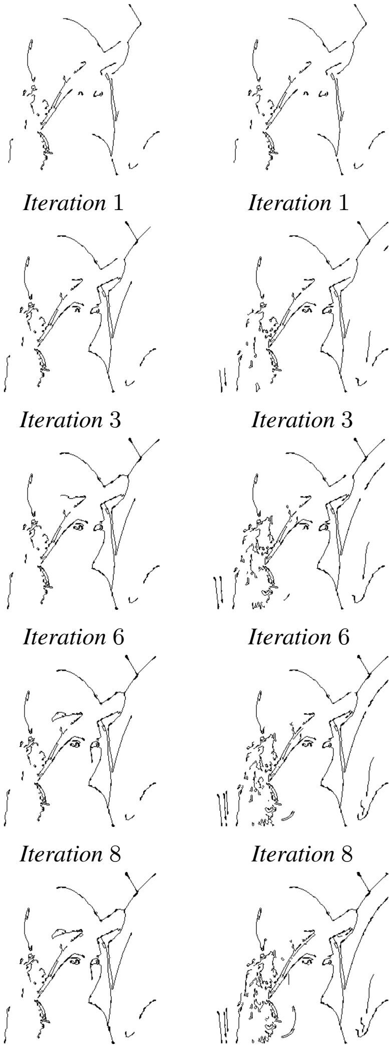
(a) Iteration 10   *(b) Iteration* 10   *(c) Iteration* 23

Figure 5 *– The binary images of the shapes image obtained over different iterations of Baterina and Oppus (2010), FACS, and the extended FACS.*

The results for the lena and the mandril images that are obtained by both FACS and the extended FACS are shown in Figure 6 and Figure 7. Since the results generated by both FACS and the Extended FACS are noise-free, we can combine the binary images obtained over several iterations in order to achieve a better edge detection.

8 Comparisons and Analysis

In this section, we perform a comparison between the three algorithms: the ACS system presented in Baterina and Oppus (2010) (without filtering), the FACS, and the Extended FACS. We perform our comparison from two points of view: the amount of pheromone deposited by ants over the different iterations (a measure of quality) and the time required by the algorithm over the different iterations (a measure of complexity). FACS deposits the highest amount of pheromone as we can see in Figure 8(a). This is due to the fact that FACS initiates the ants at the positions of highest heuristic values, thus, a large number of intersections exists between the solutions found by the different ants. Therefore, the selected pixels in the previous iterations have a higher probability to be re-selected and assigned additional amounts of pheromone. The amount of pheromone assigned by the Extended FACS is, in general, less than the amount assigned by FACS since ants could be placed at different initial positions. In cases when FACS is able to generate different



Iteration 1 *Iteration* 1

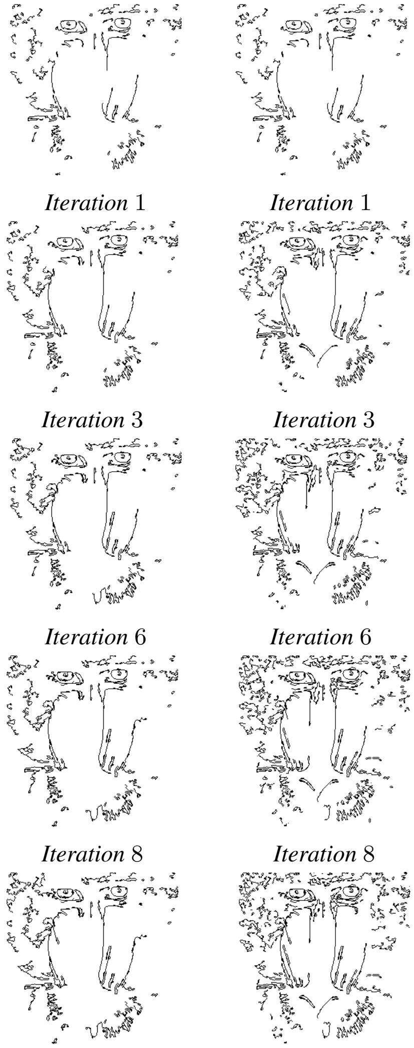
Iteration 3 *Iteration* 3

Iteration 6 *Iteration* 6

Iteration 8 *Iteration* 8

Iteration 10 *Iteration* 10

Figure 6 *– The binary images of the image ”Lena” obtained over different iterations for both FACS and the Extended FACS.*



Iteration 1 *Iteration* 1

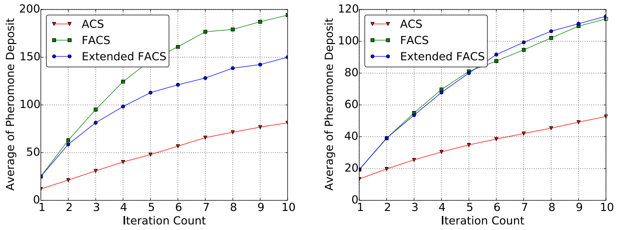
Iteration 3 *Iteration* 3

Iteration 6 *Iteration* 6

Iteration 8 *Iteration* 8

Iteration 10 *Iteration* 10

Figure 7 *– The binary images of the image ”Mandril” obtained over different iterations for both FACS and the Extended FACS.*



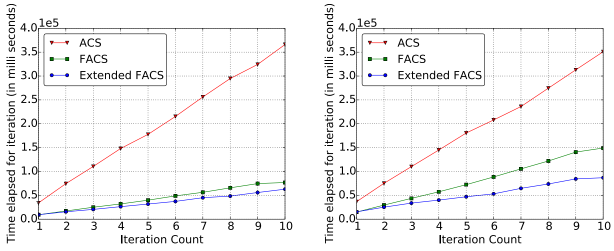
(a) *The ”Lena” image.* ***(b)*** *The ”Mandril” image.*

Figure 8 *– The average amount of pheromone deposited by ants over* 10 *iterations.*

solutions over the considered number of iterations, the average of the deposited pheromone becomes similar to the one of the Extended FACS as we can see in Figure 8(b). The average amount of deposited pheromone is at its minimum when the ants are initialized at random positions and that is what we can see in both Figure 8(a) and (b).

The second criteria we use to compare the three algorithms is the time required by the algorithm to perform the edge detection over a specific number of iterations. This represents a measure of the algorithm complexity. From Figures 9(a) and (b), we can notice that FACS and the Extended FACS perform significantly better than the ACS presented in Baterina and Oppus (2010) in terms of the required time. The reason behind is that when using FACS or the Extended FACS, the probability for ants to re-visit positions is higher. Hence, the update of the global pheromone deals with less number of positions (pixels), which leads to a faster update.

In summary, FACS and the Extended FACS offer solutions with a better quality (higher pheromone concentration) and within a significantly shorter time (lower complexity) than by traditional ACS. Moreover, they don't need any additional thresholding or filtering processes.



(a) The "Lena" image. *(b) The "Mandril" image.*

***Figure 9** – The time required for the detection of image edges over* 10 *iterations.*

9 Conclusion

In this paper, we have presented an efficient Ant Colony System (ACS) for solving the fundamental problem of edge detection which is required in a large number of applications. Two variants of the algorithm were presented: the FACS, in which the ants are initialized at positions selected based on the ranked heuristic information, and the Extended FACS in which the ants are ranked to avoid stagnation problems and generate eventually better solutions by placing the low-ranked ants at new initial positions. The proposed algorithms were compared with the ACS system presented in Baterina and Oppus (2010) without filtering.
The two algorithms proposed in this paper have outperformed the traditional ACS (without filtering) and higher quality solutions were obtained in significantly shorter time, without the need for addition noise-filtering processes.

References

Baterina, A. and Oppus, C. (2010). Ant colony optimization for image edge detection. In *Proceedings of the 9th WSEAS International Conference on Signal Processing, Robotics and Automation*, ISPRA'10, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS).

Blum, C. (2005). Beam-aco–hybridizing ant colony optimization with beam search: An application to open shop scheduling. *Computers & Operations Research*, 32(6):1565–1591.

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698.

Christian, B. (2005). Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews*, 2(4):353–373.

Colorni, A., Dorigo, M., and Maniezzo, V. (1991). Distributed optimization by ant colonies. In *Proceedings of the first European conference on artificial life*, volume 142, pages 134–142, Paris, France.

Davis, L. (1975). A survey of edge detection techniques. *Computer graphics and image processing*, 4(3):248–270.

Dorigo, M. and Gambardella, L. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66.

Dorigo, M. and Stützle, T. (2004). *Ant Colony Optimization*. Bradford Company, MA, USA.

Gelfand, I., Fomin, S., and Silverman, R. (2000). *Calculus of Variations*. Dover Books on Mathematics. Dover Publications.

Haralick, R. (1984). Digital step edges from zero crossing of second directional derivatives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(1):58–68.

Jian, Z., Jiliu, Z., Kun, H., and Huanzhou, L. (2012). Image edge detection using quantum ant colony optimization. *International Journal of Digital Content Technology and its Applications(JDCTA)*, 6(11):402–405.

Manish, T., Murugan, D., and Ganesh, K. (2013). Hybrid edge detection using canny and ant colony optimization. *Communications in Information Science and Management Engineering*, 3(8):402–405.

Marr, D. and Hildreth, E. (1980). Theory of edge detection. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 207(1167):187–217.

Martens, D., Backer, M. D., Haesen, R., Vanthienen, J., Snoeck, M., and Baesens, B. (2007). Classification with ant colony optimization. *IEEE Transactions on Evolutionary Computation*, 11(5):651–665.

Nezamabadi-pour, H., Saryazdi, S., and Rashedi, E. (2006). Edge detection using ant algorithms. *Soft Computing*, 10(7):623–628.

Prewitt, J. (1970). Object enhancement and extraction. *Picture processing and Psychopictorics*, 10(1):15–19.

Rezaee, A. (2008). Extracting edge of images with ant colony. *Journal of Electrical Engineering*, 59(1):57–59.

Secomandi, N. (2000). Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. *Computers & Operations Research*, 27(11):1201–1225.

Stützle, T. and Hoos, H. (2000). Max–min ant system. *Future generation computer systems*, 16(8):889–914.

Tian, J., Yu, W., and Xie, S. (2008). An ant colony optimization algorithm for image edge detection. In *IEEE Congress on Evolutionary Computation*, pages 751–756.

Toth, P. and Vigo, D. (2002). Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1):487–512.

Zhuang, X. (2004). Edge feature extraction in digital images with the ant colony system. In *IEEE International Conference on Computational Intelligence for Measurement Systems and Applications*, pages 133–136. IEEE.

Ziou, D. and Tabbone, S. (1998). Edge detection techniques-an overview. *Pattern Recognition and Image Analysis*, 8:537–559.

Abnormality Detection in Robots Exhibiting Composite Swarm Behaviours

Danesh Tarapore[1], Anders Lyhne Christensen[2,3] and Jon Timmis[1]

[1]York Robotics Laboratory, Department of Electronics, University of York, York, UK
[2]Bio-inspired Computation and Intelligent Machines Lab, Lisbon, Portugal
[3]Instituto de Telecomunicações, Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon, Portugal
danesh.tarapore@york.ac.uk

Abstract

Fault detection is one of the most prominent challenges in the field of multirobot systems (MRS). Most existing fault-tolerant systems prescribe a characterisation of normal behaviours (fault-free behaviours), and train a model to recognise them. Behaviours not recognised by the model are labelled abnormal. MRS employing these models do not transition well to scenarios involving gradual changes in normal behaviour. In such scenarios, existing fault-detection systems may not be applicable, or may incur potentially costly false positive detections. We propose to address this challenging problem by taking inspiration from the regulation of tolerance and (auto)immunity in the adaptive immune system. We deploy an immune system-based fault-detection approach to detect abnormalities in *heterogeneously* behaving robots. Results of extensive simulation-based experiments demonstrate that a distributed MRS can correctly tolerate delayed propagation of different normal behaviours in the collective, at low false-positive rates. Furthermore, the fault-detection system is able to reliably detect robots performing different fault-simulating behaviours.

Introduction

The field of multirobot systems (MRS) has progressed rapidly in recent years, with groups of robots performing increasingly complex behaviours, ranging from self-assembly (Christensen et al., 2008a), to warehouse-management (Wurman et al., 2007). Despite the availability of several low-cost robot platforms (IFR, 2014), and coordination algorithms for task allocation and division of labour (e.g., Berman et al. (2009)), platform reliability and endurance still inhibit the wide-spread usage of MRS outside of the laboratory (Dunbabin and Marques, 2012). The individual robots of a MRS are susceptible to failures, prominently resulting from electro-mechanical faults in the robot's sensor and actuation devices, and bugs in the software controlling the robot (Winfield and Nembrini, 2006). Consequent to the wide variety of intricate inter-robot interactions affecting robot behaviour, the prediction, detection and/or diagnosis of potential faults for an individual robot represent major challenges. While the large number of individual robots in self-organised robot collectives may produce a robust and resilient system, even in such systems, robots that exhibit partial failure have the potential to disrupt the entire collective (Bjerknes and Winfield, 2013). Explicit fault detection is therefore crucial to enhance the autonomy and operating capacity of MRS.

In most engineered fault-detection systems, robots are trained (e.g., using supervised learning) to detect anomalies in their own sensory and actuator data (e.g., Christensen et al. (2008b)). While these approaches can provide robust fault detection when trained on the normal behaviour (no faults present) of the target system, they do not transcend well to changes in this behaviour. Other fault-detection systems, specifically designed for MRS, allow the robots to detect faults in each other (e.g., Lau et al. (2011)). However, many of these approaches are centralised and/or rely on detailed knowledge of the MRS tasks. Furthermore, the more task-generic and decentralised versions of such approaches are limited to detecting robots experiencing complete failure (details in the Related Work section).

An interesting analogy can be made between task-generic fault-detection systems and the adaptive immune system in vertebrates (e.g., Timmis et al. (2010); Tarapore et al. (2012)). The immune system acts to help defend and repair the body's cells and tissues in response to pathogenic insults, has the need to differentiate between what might be normal, that is, normally functioning cells and tissues, and abnormalities such as invading pathogen (Janeway et al., 1997). The characteristics of these abnormalities are in principle open-ended, and therefore differ from the limited set of faults the current approaches to robot fault-detection are designed to detect. Experimental evidence indicates that the tolerance exhibited by the immune system results from the dynamics and interactions between specific regulatory and effector T-cells (e.g., Sakaguchi (2004)). The decentralised nature of these intercellular interactions imparts a high degree of robustness within the system, and a flexibility to respond to a very broad range of possible pathogenic attacks. Such properties provide rich inspiration for the engineering of robust, fault-tolerant systems.

In previous work, we developed a generic fault-detection approach, based on the adaptive immune system, for exogenous fault detection in large-scale MRS (Tarapore et al., 2015). An agent-based simulator was used to model scenarios where individual robots have to tolerate certain behaviours, while mounting an immune response against others. The salient feature of the developed fault-detection system was that the characterisation of behaviour (normal or faulty) were not prescribed apriori, but rather learned online based on their abundance in the MRS. Behaviours exhibited by many of the robots in the MRS were considered normal. By contrast, rare behaviours exhibited by a single or few robots were detected as abnormal behaviours that could be caused by a fault. The resultant fault-detection system was successfully utilised in MRS comprising of *homogeneously* behaving robots. However, in many MRS, the individual robots of the collective often exhibit distinct behaviours at any given time. For example, even in simple foraging scenarios, the different behaviours exhibited may include, searching for resources, signalling the presence of new resources, and returning them to the nest. The scenarios are often more complex, and include composite exploration and exploitation like behaviours. In such tasks, existing adaptive fault-detection systems may result in a large number of false-positive classifications of normal behaviours.

In this paper, we propose a solution to the above issue, specifically: extending our generic exogenous fault-detection system to operate successfully on *heterogeneously* behaving robots of an MRS. In our extended model, a history of past behaviour observations is taken into account in order to classify robot behaviour, not only as normal/abnormal, but also as *suspicious*. Using a history of past T-cell populations embodied on the robot, observed behaviours misclassified as abnormal in our original model are now simply treated as suspicious, if the behaviour was considered normal in the past. We demonstrate the capacity of the system to tolerate normal behaviours, despite temporal variations – ranging from an almost simultaneous to slow changes in the behaviour across the MRS. Furthermore, our extended fault-detection system continues to reliably detect abnormally behaving robots.

The rest of the paper is organised as follows: in the following section, we describe the different approaches to fault detection in autonomous robots, followed by our fault-detection system based on the adaptive immune system. We then present extensions to our fault-detection system for heterogeneously behaving robots. We go on to report the results of our experiments in different scenarios, and under varying behaviour transition rates. Finally, we discuss our approach to fault detection and highlight the conclusions of this study.

Related work

The engineering of fault-detection systems for robots is a well-studied problem, and can be broadly classified into *endogenous* and *exogenous* approaches. The endogenous fault detection approaches have robots proprioceptively detecting faults in their individual behaviour (e.g., Christensen et al. (2008b); Skoundrianos and Tzafestas (2004)). The models assume that hardware faults affect the flow of sensory information, and actuation of the robot. Consequently, statistical learning algorithms (e.g., artificial neural networks) are trained to detect anomalies based on the input-output relationship of focal components on the robot. These approaches have been successfully used to detect faults in components such as, wheels (Skoundrianos and Tzafestas, 2004), and tracks with wheels (Christensen et al., 2008b) of a mobile robot. Most endogenous fault-detection models are built on the assumption that the normal operating behaviour of the robot is known, and can be characterised pre-deployment. Consequently, the models are trained to recognise prescribed normal behaviour, and behaviours not recognised by the model are labelled abnormal. However, while such approaches have resulted in reliable fault detection in specific scenarios, they may not easily transition to different and varying characterisations of normality in MRS (e.g., online adaptation of existing behaviours or even learning of new behaviours). In summary, endogenous fault-detection systems are finely tuned to the particular behaviour of the target system, under a specific set of task parameters.

Exogenous fault detection systems leverage the multitude of robots constituting a MRS, to provide individual robots the capability to detect faults in one another (e.g., Parker (1998); Christensen et al. (2009); Millard et al. (2014); Lau et al. (2011)). Such an approach is particularly advantageous to detect faults that are difficult to detect endogenously by the robot (e.g., mechanical failures consequent to an unstable connection to a power source), or that disable the robots communication and capability to alert other robots or a human operator. However, while exogenous approaches do provide some interesting results of robust fault detection and tolerance (e.g., see ALLIANCE software architecture Parker (1998)), successful fault detection require prior knowledge of the various tasks to be performed, and their corresponding measures of performance. Furthermore, many of these fault-detection approaches (e.g., Gerkey and Matarić (2002); Parker (1998)) are designed for MRS consisting of a limited number of tightly coupled, and relatively complex robots.

Adaptive immune system based fault-detection

Our previously developed fault-detection system for homogeneously behaving robots is based on the *crossregulation model* (CRM) (Leon et al., 2000), a mathematical model that captures the robust maintenance of immunological tolerance by allowing the system to discriminate between antigens based solely on their density and persistence in the environment. According to Leon et al. (2000), the immune system is able to tolerate body antigens (the molecular components of body tissues) that are characteristically persistent and abundant, and to mount an immune response to foreign pathogens, that are characterised as being neither persistent nor abundant.

The CRM describes the population dynamics of cells of the adaptive immune system, consisting of three mutually interacting cell types: (a) Antigen presenting cells (APCs) that present the antigen on their surface. Individual APCs have a fixed number of binding sites on which effector and regulatory cells can form conjugates; (b) effector cells $\mathtt{T}_\mathtt{E}$ that can potentially mount immune responses which, depending on receptor specificity, may be directed to foreign pathogens or to body-antigens; and (c) regulatory cells $\mathtt{T}_\mathtt{R}$ that suppress proliferation of $\mathtt{T}_\mathtt{E}$ cells with similar specificities. Furthermore, APCs are classified into different sub-populations of equivalent APCs, with each APC in a sub-population presenting the same antigen on its surface. Similarly, effector and regulatory cells are also classified into different clones according to their specificity.

A mathematical formulation comprising ordinary differential equations, of the dynamics of interactions between effector cells and regulatory cells, with APCs, is detailed in Tarapore et al. (2015). In the next subsection, we provide an overview of these interactions, introduce the important parameters, and highlight the interesting properties of the CRM (detailed description of model at Carneiro et al. (2007)). We then describe the implementation of the CRM in the MRS, and the resultant fault detection achieved by the model.

Functioning of the CRM

The CRM provides a differential equation governing each of the clonal types (i) of effector (E_i), and regulatory (R_i) T-cells. The sub-populations of each of these clonal types is subject to the following: (a) growth by proliferation (division of parent cells to two daughter cells) of their individual activated cells; and (b) shrinkage consequent to death of T-cells.

The density of proliferating T-cells of each clonal type i, is dependent on their interactions with APCs of each sub-populations j. Consider the interactions between the i-th T-cell clone and the j-th APC population. The resulting conjugates C_{ij} are subject to the following: (a) Formation of new conjugates by the free T-cells of clone i with available binding sites on APCs of sub-population j. This conjugation rate is also controlled by the affinity between the T-cells clone i and APCs sub-population j; and (b) Dissociation of existing conjugated T-cells from APCs. The density of activated effector and regulatory cells is computed from the quasi-steady state densities of the conjugates. The conjugated effector cells proliferate in the absence of regulatory cells on the same APC. In contrast, conjugated regulatory cells can only proliferate if at least one effector cell is simultaneously conjugated to the same APC.

Table 1: Parameters of the CRM implementation.

Param.	Description	Value
l	Length of binary feature vector	6 bits
M	Maximum number of different feature vectors	2^l
N	Maximum number of T-cell clones	2^l
c	Cross-reactivity between T-cells and APCs	0.15
I_E	Density of new effector cells introduced at each simulation time-step	10 a.u.
I_R	Density of new regulatory cells introduced at each simulation time-step	10 a.u.
k	Feature vectors to APCs scaling factor	0.002
S	Length of time CRM instance is numerically integrated, in a single robot control cycle	5×10^7 a.u.
d	Proportion of T-cells diffused to neighbouring robots	0.5
s_j	Suspicion value associated with feature vector FV_j	$-$
Δ_s	Increment to suspicion value for newly observed feature vector	0.95
γ	Threshold below which feature-vector is interpreted as suspicious and not abnormal	0.95

Execution of the CRM on a robot

Our original CRM-based model implemented on a distributed embodied MRS, in simulation, affords the system the capacity to detect abnormally behaving robots, whilst maintaining a tolerance towards normal robot behaviour (see Tarapore et al. (2015)). Within the CRM framework, behaviours exhibited by an abundant proportion of the robots (normal behaviour) are interpreted as body antigens (so normal). By contrast, faulty or abnormal behaviours are considered foreign antigens (abnormal). Each robot executes an independent instance of the CRM.

The CRM-based fault-detection model was tested for four normal swarm behaviours (aggregation, flocking, dispersion, and homing) and four fault-simulating behaviours. The fault-simulating behaviours performed by one of the 20 robots (selected at random) was, (a) move continually in a straightly line (STRLN); (b) perform a random walk, with a 0.01 probability of changing to a new random direction each simulation time-step (RNDWK); (c) circle with diameter 1 unit around a fixed point (CIRCLE); or (d) stop completely (STOP). These additional behaviors were introduced to mimic: (a) software bugs and sensor faults in the robot controller (STRLN and RNDWK); (b) motor malfunctions (CIRCLE); and (c) a broken or dead battery (STOP).

In both the original, and extended CRM-based models, a robot computes a 6 bit binary feature vector (concatenation of 6 simple Boolean features) encoding its behaviour (Table 2), at the start of each control cycle. The robot then senses the feature vectors of its 10 nearest neighbours (tested with up to 100 simulated robots in MRS, see Tarapore et al. (2015)), and counts the number of robots assigned to each of the 2^6 feature vectors (FV_j, $j \in \{1 \ldots 2^6\}$). In an individ-

Table 2: Boolean features encoding robot behaviour (parameters in Table 3).

Notation	Value at time τ*
$F_1(\tau)$	$\frac{\sum_{t=\tau}^{\tau-W} U[n_i(t)]}{W} > 0.5$
$F_2(\tau)$	$\frac{\sum_{t=\tau}^{\tau-W} U[n_o(t)]}{W} > 0.5$
$F_3(\tau)$	$p(\tau) > 0.05W\lvert\vec{v}_{\mathtt{max}}\rvert$
$F_4(\tau)$	$\lvert\vec{v}(\tau)\rvert > 0.05\lvert\vec{v}_{\mathtt{max}}\rvert$
$F_5(\tau)$	$\sum_{t=\tau}^{\tau-W} U[n_i(\tau) + n_o(\tau)] \wedge U[\omega'(\tau) - 0.03\omega'_{\mathtt{max}}] > 0$
$F_6(\tau)$	$\sum_{t=\tau}^{\tau-W} \neg U[n_i(\tau) + n_o(\tau)] \wedge U[\omega'(\tau) - 0.03\omega'_{\mathtt{max}}] > 0$

*The Boolean feature is set if the condition is satisfied, else 0. Function $U[x]$ is 1 if $x > 0$, and 0 otherwise.

ual robot's internal CRM instance, APCs are then generated corresponding to each of the feature vectors perceived. Each APC presents an individual feature vector to the T-cells. The number of each type of the APCs generated $A_j = kFV_j$, for $j \in \{1, \ldots, M\}$, where k is a scaling constant, and M is the number of different feature vectors perceived by the robot. The T-cell clones $(T_1, T_2, \ldots, T_N)$, each have a different receptor encoded as a binary string, which determines their affinity to the APC population. The affinity between T-cell clonal i and APC population j is denoted by θ_{ij}:

$$\theta_{ij} = \exp\left(-\frac{H[i,j]}{cl}\right) \tag{1}$$

where H is the Hamming distance between the receptor of T_i and the feature vector presented by A_j, l is the length of the presented feature vector, and c is the cross-reactivity between T-cells and APCs.

At the start of the simulation, the number of effector and regulator cells on each robot is initialised to I_E and I_R, respectively. Following this, Algorithm 1 (parameters in Table 1) is performed by every robot in each control cycle, allowing the robots to execute their internal CRM. The robots begin by sensing their neighbours, and then compute the distribution of feature vectors. The CRM is then numerically integrated for time S, allowing the system to respond to the different APCs. After computing the number of effector and regulatory cells at time S, the cells diffuse among robots. In this communication phase, each robot selects at random (linear distribution weighted on total T-cell density in the CRM instance) another robot, from one of its 10 nearest neighbours. Following the selection, each robot sends and receives d of its effector and regulatory cells. Finally, the robot decides the nature of each feature vector FV_j sensed by first computing the following quantities:

$$E = \sum_{i=1}^{N} \theta_{ij} E_i \qquad R = \sum_{i=1}^{N} \theta_{ij} R_i \tag{2}$$

and tolerating the feature vector if $R > E$. By contrast, if $E > R$, the feature vector is classified as faulty by the robot.

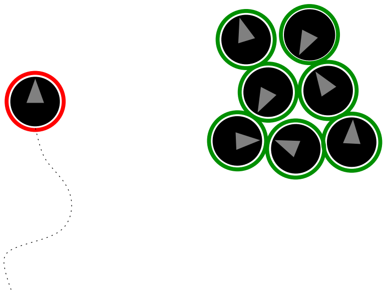


Figure 1: Example of a robot (left) unable to join an aggregate (right) due to faulty sensors. This robot is detected as behaving abnormally (coloured red) by its neighbours, whereas the rest of the swarm is behaving normally and tolerated (coloured green).

The CRM deployed in the MRS is passive and does not alter the behaviour of the robots. Rather, the individual robots merely report the outcome of the classification for the different behaviors observed in their vicinity, at each simulation time step. At the end of each time step, a robot's behaviour is considered normal, if a simple majority of the robot's 10 nearest neighbours tolerate it. Similarly, the behaviour is treated abnormal, if a simple majority of these neighbours interpret it as faulty.

Extending fault-detection to heterogeneously behaving robots

The CRM-based abnormality detection system classifies behaviours solely based on their abundance in the MRS. Behaviours exhibited by many, or the majority of the robots of the MRS are considered as normal. By contrast, rare behaviours exhibited by one of few robots are considered as abnormal, and assumed to be resulting from a fault on the robot. While such an approach is capable of robustly tolerating normal behaviours, and reliably detecting faults in homogeneously behaving robots, it is not designed to detect faults in robots executing complex (or composite) behaviours, or in which behaviour transitions propagate gradually across the MRS. In such scenarios, normal behaviour exhibited by a minority of robots of the MRS trigger false positive incidents.

In order to extend our CRM-based model to scenarios in which robots independently perform different behaviours at different times, behaviour classification must be based on observations made over a period of time. In accounting for past observations, we may ask if a behaviour detected as abnormal in the current time step, has always been abnormal? Considering the behaviour was also abnormal in the previous contexts that it was observed (context is the behaviours of the rest of the MRS), than it may indeed be abnormal. However, if the behaviour was treated as normal in the past, or if a new behaviour has just emerged in the

MRS and was therefore not encountered in the past, we may not want to classify it as abnormal (and take appropriate action), but merely treat such behaviour as *suspicious*. This suspicion associated with an observed behaviour quantifies the proportion of past contexts, with respect to which the presently observed behaviour would have been considered as abnormal.

Algorithm 1 A robot's control loop (simulation of a CRM instance).

```
Compute distribution of feature vectors ($(FV_j)$) of neighbouring robots
Assign feature vectors to APCs i.e., $\forall j$, $A_j = kFV_j$
$\forall j \in \{1, 2 \ldots M\}$, if $A_j > 0$, increment $E_j$ and $R_j$ by $I_E$ and $I_R$, respectively
while $time \leq S$ do
    $\forall i \in \{1, 2 \ldots N\}$ and $T_i > 0$, and $\forall j \in \{1, 2 \ldots M\}$ where $A_j > 0$, compute the number of conjugated cells $C_{ij}$ in quasi-steady state.
    Using the number of conjugated cells, compute the updated number of effector and regulatory cells with the Euler-Heun adaptive step method (Butcher, 2003).
    Increment $time$
end while
Randomly select one of the robots in the communication range following a linear distribution and weighted by the total number of cells on the respective neighbouring robots
Exchange cells with robot
For each feature vector, compute the sum of effector and regulatory cells, weighted by their affinity.
Tolerate the feature vector if total regulatory cells exceeds effectors, else interpret it as faulty.
```

The suspicion is computed by the robot, when its instance of the CRM classifies an observed behaviour as abnormal. Algorithm 2 details the procedure to compute the suspicion (s_j) associated with a feature vector FV_j, already classified as abnormal. It involves an increment to the suspicion value for every simulation time-step t over the recorded history D, when FV_j would have been considered abnormal (parameters in Table 1).

$$E^t = \sum_{i=1}^{N} \theta_{ij} E_i^t \qquad R^t = \sum_{i=1}^{N} \theta_{ij} R_i^t \tag{3}$$

where E_i^t and R_i^t are the recorded density of effector and regulatory cells at simulation time-step t.

When $E^t > R^t$, the FV_j is considered abnormal with respect to the context at simulation time-step t, and the suspicion s_j associated with FV_j is incremented by 1.

The appearance of a new behaviour, i.e., a behaviour not observed by the robot at time-step t, also results in an increment to s_j, but by a smaller value of Δ_s ($\Delta_s < 1$).

Finally, the feature vector FV_j is considered as *abnormal* if its normalised suspicion value (s_j/D) exceeds a threshold γ. Otherwise, FV_j is merely considered as *suspicious*.

Latency in fault detection: The minimisation of the time required to detect a behaviour as abnormal since it began to be executed is an important requirement of any fault-detection system. At the offset of a new behaviour, our suspicion towards the behaviour is incremented by a small value (Δ_s), since we may not want to immediately classify it as abnormal. Obviously, the suspicion parameter Δ_s influences detection latency, and can be analysed. For a behaviour to be detected as abnormal in our extended model, the following condition has to be satisfied:

Algorithm 2 Subroutine to determine suspicion, called when feature vector FV_j is interpreted as abnormal.

```
{Initialise suspicion value associated to $FV_j$}
$s_j \leftarrow 0$
{Iterate over past $D$ time-steps}
for $t = current$ to $current - D$ do
    {If $FV_j$ was not observed at time $t$, increment $s_j$ by a lower value}
    if $FV_j$ not observed at time $t$ then
        $s_j \leftarrow s_j + \Delta_s$
    else
        {Analyse $FV_j$ with T-cell population of time $t$}
        $E(t) \leftarrow \sum_{i=1}^{N} \theta_{ij} E_i(t)$
        $R(t) \leftarrow \sum_{i=1}^{N} \theta_{ij} R_i(t)$
        {If $FV_j$ would have been interpreted as abnormal at time $t$, increment $s_j$ by a higher value}
        if $E(t) > R(t)$ then
            $s_j \leftarrow s_j + 1$
        end if
    end if
end for
{Compare normalised suspicion value to threshold}
if $s_j/D < \gamma$ then
    $FV_j$ is suspicious
else
    $FV_j$ is abnormal
end if
```

$$\frac{y}{D} + \Delta_s \left(\frac{D - y}{D} \right) \geq \gamma \tag{4}$$

where y is the amount of time in the past that the behaviour would have been classified as abnormal (see eq. 3).

Solving eq. 4 for y with parameters in Table 1, the latency is at least 295 s, 128.3 s, and 45 s, for Δ_s at 0.90, 0.94, and 0.95, respectively (including time window W to formulate the feature vector). The latency may be longer if the discriminatory features in the feature vector are not present in time window W.

Experiments

Experimental setup: We conducted a series of experiments to assess the performance of our abnormality detection approach. In all experiments, we simulated a swarm of 20 e-puck like robots located in a 5×5 m^2 toroidal environment. Each robot had a diameter of 7.5 cm and a maximum speed

Table 3: Parameters of simulated robot

Param.	Description	Value
$\lvert\vec{v}_{\mathtt{max}}\rvert$	Maximum linear speed of robot	10 cm/s
$\lvert\vec{v}\rvert$	Linear speed of robot	–
$\omega_{\mathtt{max}}$	Maximum change in direction of robot per control cycle	π radians
ω	Change in direction of robot per control cycle	–
n_i	Number of neighbouring robots in the inner range of $[0, 30]$ cm	–
n_o	Number of neighbouring robots in the outer range of $(30, 60]$ cm	–
W	Length of the time window for feature computation	45 s
p	Distance traversed by the robot in the past W s	–

of 10 cm/s (see Table 3).[1] At the start of each experiment, the robots were placed at a random location and assigned a random orientation drawn from uniform distributions.

Latency in fault detection: In a first set of experiments, we assessed the capacity of our CRM-based detector to correctly detect abnormally behaving robots. All robots initially performed Dispersion in which they tried to maximise the distance to all neighbours. Halfway through each experiment, one of the robots in the swarm simulated a fault by switching to the STOP behaviour, which caused the robot to remain immobile. We conducted 20 replicates of each of three experimental setups with different values of the suspicious increment parameter, Δ_s: 0.90, 0.94, and 0.95. Each experimental replicate had a duration of $1,500$ seconds of simulated time. In all 20 out of 20 replicates, for all values of Δ_s, we observed that the fault-simulating robot was correctly detected as behaving abnormally by other members of the swarm. However, the latency, that is the time between a fault-simulating robot begins to execute STOP behaviour, and until it is detected, varied significantly across the three setups (Mann-Whitney test, $df = 18$, all $p < 0.001$, see Fig. 2). For $\Delta_s = 0.90$, the median fault detection latency was 288.2 ± 23.6 s (Median $\pm$ IQR), for $\Delta_s = 0.94$ the median latency was 108.8 ± 20.7 s, while for $\Delta_s = 0.95$ the median latency was 33.2 ± 21 s. As expected (see eq. 4), with suspicion parameter Δ_s at 0.95, the MRS achieved the lowest latency in detecting simulated faults.

Using Δ_s at 0.95, we also tried the following normal/abnormal behaviour combinations in the same setup: (i) Aggregation/Dispersion; (ii) Flocking/RNDWK; and (iii) Homing/Dispersion (see Table 4). In the three behaviour combinations, the abnormally behaving robot was detected in less than 3.5 minutes (median). However, for the Aggregation/Dispersion and Homing/Dispersion behaviours, the abnormal robot exhibiting Dispersion was not detected in 8 of 20, and 7 of 20 replicates, respectively. In these replicates, the abnormal robot was trapped by other aggregating or homing robots when it switched to performing Dispersion, and therefore could not be differentiated from these robots.

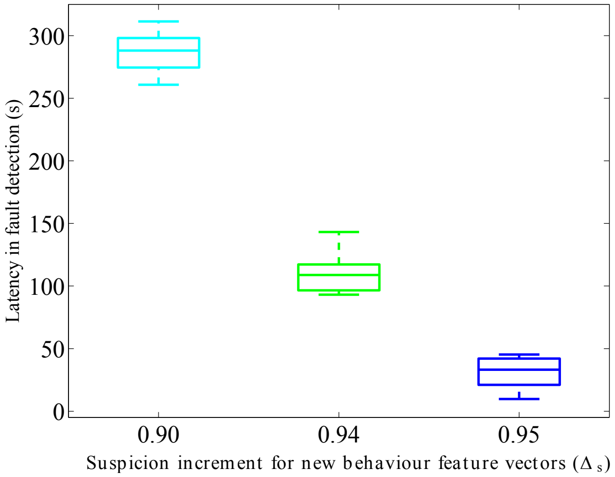


Figure 2: **Latency in the detection of fault simulating behaviour.** Time required to detect an robot performing fault simulating behaviour across 20 replicates for different suspicion increment parameters. The detected robot switches its behaviour from dispersion (normal behaviour of the MRS) to stop halfway through the simulation.

False-positive incidents for heterogeneously behaving robots: The minimisation of false positives is important in fault-detection systems, as subsequent fault accommodation may be costly and could lead to the exclusion of capable robots. In a series of experiments, we assessed the system's capacity to avoid false positives when transitions in behaviour occur over time. We conducted 20 replicates in which all robots switched behaviour. In each replicate, the robots started out by performing Dispersion, then gradually switched to the STOP behaviour over a period of time, and then reverted back to Dispersion over a period of time. Each experimental replicate had a duration of $1,500$ seconds. During the first 500 seconds, all the robots of the MRS performed Dispersion. Then, robots selected at random (following a uniform distribution), began switching to the STOP behaviour. The time between robots switching behaviour was 3.75 s, 6.25 s, 12.5 s, and 25.0 s (in four separate and independent experimental setups). After all robot had switched behaviour, robots (selected at random) began to switch back to Dispersion with the same delay used in the initial switch. It should be noted that while in the previous set of experiments (Fig. 2), the STOP behaviour was used to simulate a fault, in this new set of experiments, the STOP behaviour should no longer be considered abnormal since all the robots in the swarm eventually transition to this

[1] Simulation source code can be downloaded from `https://github.com/daneshtarapore/robotsim_ecal2015.git`

Table 4: Number of false-positive incidents per robot, and the latency, for different behaviours propagated in the MRS. Suspicion parameter Δ_s is 0.95.

Behaviours	Number of false-positives incidents at behaviour switching time				Latency (s)
	3.75 s	6.25 s	12.5 s	25 s	
Dispersion, STOP	8.1 ± 12.6	14.8 ± 22.6	38.0 ± 30.7	83.0 ± 22.4	33.2 ± 21.0
Aggregation, Dispersion	2.3 ± 7.0	3.5 ± 9.7	5.1 ± 13.1	21.5 ± 25.9	49.9 ± 28.0†
Flocking, RNDWK	9.1 ± 15.7	17.8 ± 17.6	19.9 ± 22.1	18.8 ± 27.6	194.7 ± 219.4
Homing, Dispersion	0.4 ± 5.5	0.2 ± 7.2	7.7 ± 9.3	4.7 ± 10.6	88.2 ± 67.7‡

†In Aggregation/Dispersion, the dispersing robot was not detected as abnormal in 8 of 20 replicates.

‡In Homing/Dispersion, the dispersing robot was not detected as abnormal in 7 of 20 replicates.

behaviour. The results, in terms of number of false-positive incidents, for the original (suspicion disabled) and the extended CRM-based fault detector (suspicion enabled), and for different behaviour transition periods are shown in Fig. 3.

Our extended CRM-based model achieved at least an order of magnitude improvement in the number of false-positive incidents over the original model, for all four behaviour switching times. The difference in performance was higher for the more gradual transitions in behaviour. For the extended CRM-based model, at $\Delta_s = 0.90$, the number of false-positive incidents incurred by the MRS was not affected by the behaviour switching delay. However, for $\Delta_s = 0.95$, the behaviour switching delay significantly affected the number of false-positive incidents (Kruskal-Wallis test: $p < 0.001$). Experiments where the MRS was subjected to a higher behaviour switching delay between robots of 25 s incurred more false-positive classifications than MRS experiencing behaviour-switching delays of 3.75 s, 6.25 s and 12.5 s (Mann-Whitney test, $df = 18$, all $p < 0.001$), but with the difference not exceeding 75 incidents in a total of 15,000 control cycles.

In our extended CRM-based model, a trade-off exists between latency and the number of false positives, regulated by the suspicion parameter Δ_s. For Δ_s at 0.90, the model registered a low number of false positives, but a high latency in detecting fault-simulating robots (Fig. 2 and 3). Incrementing Δ_s to 0.95 resulted in a much lower latency, and only a slight increase in the number of false positives. Using this suspicion parameter at 0.95, we also tried the following behaviour propagations across the MRS, each replicated 20 times: (i) Aggregation–Dispersion–Aggregation; (ii) Flocking–RNDWK–Flocking; and (iii) Homing–Dispersion–Homing (see Table 4). In all three behaviour combinations, the number of false-positive incidents per robot was no higher than 22 (median across 20 replicates) in $15,000$ control cycles, and irrespective of the tested behaviour switching times.

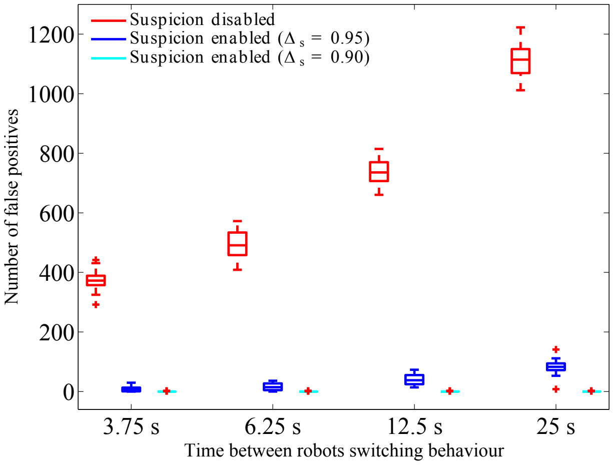


Figure 3: **False-positive incidents of fault-detection for heterogeneously behaving robots.** Number of false-positive incidents per robot across 20 replicates for four different behaviour switching times (for Dispersion–STOP–Dispersion), in each of the following three fault-detection experiments: (i) suspicion disabled (our original fault-detection model); (ii) suspicion enabled, and incremented by 0.90 ($\Delta_s = 0.90$) for new behaviour feature vectors; and (iii) suspicion enabled, and incremented by 0.95 ($\Delta_s = 0.95$) for new behaviour feature vectors.

Conclusions and future work

The results demonstrate the suitability of our generic fault-detection system for MRS exhibiting composite swam behaviours. In our approach, the individual robots of the MRS utilise a record of past behaviour observations to make more accurate normal/abnormal classifications of the behaviours of neighbouring robots. While our approach relies on an immune system-based model to operate in scenarios involving a gradual transition in MRS behaviour, other more traditional methods, such as statistical hypothesis testing could also be applied to detect changes in distribution of past sampled observations (Ladi et al., 2014). In future work, we plan to investigate how these more centralised classification approaches could be integrated into our distributed fault-detection system.

Noise evident in real sensor and actuator readings only has an impact on our fault-detection system if it can cause changes in one or more features of the feature vector. One approach to compensate for perturbations in feature vectors is to account for the specific hardware platform in the design of individual features. Additionally, the value of individual features is calculated based on several observations made over a period of time (W s), and the decision on whether to classify a faulty robot is based on votes from several robots.

There are thus several mechanisms in place to avoid misclassification, when noise is added to our simulation as part of future work.

Our fault-detection system is designed to be generic with respect to the behaviours of the underlying MRS. Behaviours are classified (normal/abnormal) solely based on their persistence and abundance in the MRS. However, a number of behaviours may be known in advance to be abnormal, or normal, in particular contexts. "Vaccinating" our immune system-based model with this information, may expedite the process of fault detection and improve the performance of the suspicion module. The long-term goal is to assess our fault-detection system in more complex behaviours for realistic task scenarios.

Acknowledgements

D.T. is supported by a Marie Curie Intra-European Fellowship (Project: GiFteD-MrS, EC Grant No. 623620). A.C. is supported by Fundação para a Ciência e a Tecnologia (FCT) under the grants EXPL/EEI-AUT/0329/2013 and UID/EEA/50008/2013. J.T. is supported by The Royal Society and The Royal Academy of Engineering.

References

Berman, S., Halasz, A., Hsieh, M. A., and Kumar, V. (2009). Optimized stochastic policies for task allocation in swarms of robots. *Robotics, IEEE Transactions on*, 25(4):927–937.

Bjerknes, J. D. and Winfield, A. F. T. (2013). On fault tolerance and scalability of swarm robotic systems. In *Distributed Autonomous Robotic Systems*, pages 431–444. Springer, Berlin Heidelberg, Germany.

Butcher, J. (2003). *Numerical methods for ordinary differential equations*, chapter 23. John Wiley & Sons, West Sussex, England, second edition.

Carneiro, J., Leon, K., Caramalho, I., Van Den Dool, C., Gardner, R., Oliveira, V., Bergman, M., Sepúlveda, N., Paixão, T., Faro, J., and Demengeot, J. (2007). When three is not a crowd: a crossregulation model of the dynamics and repertoire selection of regulatory CD4$^+$ T cells. *Immunological Reviews*, 216(1):48–68.

Christensen, A., O'Grady, R., and Dorigo, M. (2008a). SWARMORPH-script: a language for arbitrary morphology generation in self-assembling robots. *Swarm Intelligence*, 2(2-4):143–165.

Christensen, A. L., O'Grady, R., Birattari, M., and Dorigo, M. (2008b). Fault detection in autonomous robots based on fault injection and learning. *Autonomous Robots*, 24(1):49–67.

Christensen, A. L., O'Grady, R., and Dorigo, M. (2009). From fireflies to fault tolerant swarms of robots. *IEEE Transactions on Evolutionary Computation*, 13(4):754–766.

Dunbabin, M. and Marques, L. (2012). Robots for environmental monitoring: Significant advancements and applications. *IEEE Robotics & Automation Magazine*, 19(1):24–39.

Gerkey, B. P. and Matarić, M. J. (2002). Sold!: Auction methods for multirobot coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768.

IFR (2014). Industrial robot statistics. `http://www.ifr.org/industrial-robots/statistics`.

Janeway, C., Travers, P., Walport, M., and Shlomchik, M. (1997). *Immunobiology: The Immune System in Health and Disease*. Garland Science, New York, NY.

Ladi, A., Timmis, J., Tyrrell, A. M., and Hickey, P. J. (2014). Statistical hypothesis testing for chemical detection in changing environments. In *2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, pages 77–84. IEEE Press, Piscataway, NJ.

Lau, H., Bate, I., Cairns, P., and Timmis, J. (2011). Adaptive data-driven error detection in swarm robotics with statistical classifiers. *Robotics and Autonomous Systems*, 59(12):1021–1035.

Leon, K., Perez, P., Lage, A., Farob, J., and Carneiro, J. (2000). Modelling T-cell-mediated suppression dependent on interactions in multicellular conjugates. *Journal of Theoretical Biology*, 207(2):231–254.

Millard, A., Timmis, J., and Winfield, A. (2014). Run-time detection of faults in autonomous mobile robots based on the comparison of simulated and real robot behaviour. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*, pages 3720–3725. IEEE Press, Piscataway, NJ.

Parker, L. (1998). Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

Sakaguchi, S. (2004). Naturally arising CD4$^+$ regulatory T cells for immunologic self-tolerance and negative control of immune responses. *Annual Review of Immunology*, 22:531–562.

Skoundrianos, E. N. and Tzafestas, S. G. (2004). Finding fault-fault diagnosis on the wheels of a mobile robot using local model neural networks. *IEEE Robotics & Automation Magazine*, 11(3):83–90.

Tarapore, D., Christensen, A. L., Lima, P. U., and Carneiro, J. (2012). Clonal expansion without self-replicating entities. In *Proceedings of the International Conference on Artificial Immune Systems*, pages 191–204. Springer, Berlin, Germany.

Tarapore, D., Lima, P., Carneiro, J., and Christensen, A. L. (2015). To err is robotic, to tolerate immunological: fault detection in multirobot systems. *Bioinspiration & Biomemetics*, 10(1):016014.

Timmis, J., Andrews, P., and Hart, E. (2010). On artificial immune systems and swarm intelligence. *Swarm Intelligence*, 4(4):247–273.

Winfield, A. F. and Nembrini, J. (2006). Safety in numbers: fault-tolerance in robot swarms. *International Journal of Modelling, Identification and Control*, 1(1):30–37.

Wurman, P. R., D'Andrea, R., and Mountz, M. (2007). Coordinating hundreds of cooperative, autonomous vehicles in warehouses. In *Proceedings of the National Conference on Innovative Applications of Artificial Intelligence*, pages 1752–1759. AAAI Press.

The Overshoot Curriculum: Artificial Life, Education and the Human Predicament

Barry McMullin

Dublin City University, Dublin 9, Ireland
barry.mcmullin@dcu.ie

Abstract

It is well established in the scientific literature that global human civilization is in serious ecological trouble. The most comprehensive survey is perhaps that of the *Planetary Boundaries* framework (Rockström et al., 2009). The unfolding of these challenges will, of course, be a very complex process; and some detailed impacts are certainly still open to significant human management and moderation. Nonetheless, it seems clear that we are no longer dealing with a "problem", or even set of "problems", that might be "solved"; rather, this is a *predicament* — an uncertain, dynamic, and at least partially chaotic, disruption in global human development (Gilding, 2012). A predicament calls not for "solution", but for engagement, and continuous, long term, refinement of response. The purpose of this contribution is to propose a particular educational (curricular and pedagogic) response: one that specifically draws on the tools, techniques and understandings of the field of Artificial Life.

In recent decades, the mission of university education, at least in public universities, has become progressively identified simply with the direct support of economic development in its sponsoring regional community. That is, its primary role is to provide graduates with just the knowledge and skills most immediately aligned with the preceived needs of the regional economy. There is a perfectly clear logic and rationale to this development; but in a world faced with global ecological disruption, *within the lifetimes of current students*, this is neither an honest nor even an effective preparation for the challenges which they will face. Moreover, given that even limited moderation of the impending impacts of ecological limits will rely on the widest societal understanding and deeply informed and engaged leadership at all levels, it is arguable that universities have an immediate, and potentially decisive role to play in communicating the realities of the current global ecological situation.

And so, to Artificial Life; or more precisely, to Artificial Ecology. The use of computational tools to model complex biological, evolutionary, ecological and social dynamics is a foundational technique in the ALife field. Indeed, computational thinking and modelling was at the heart of the *systems dynamics* approach to socio-ecological modelling pioneered by Forrester (1982). This provided the basis for the famous (or infamous?) *Limits to Growth* (LTG) project of the Club of Rome (Meadows et al., 1972). This was the first substantive attempt to computationally model the socio-ecological dynamics of global human society and assess whether ecological impacts would be likely to limit the growth of human material activities within any practically foreseeable timeframe. While the model was necessarily crude, the robust result was that — in the absence of effective control measures to the contrary — serious limits would become apparent within the first half of the 21st century. In the 40 years since, the world has tracked remarkably close to the "standard run" of the LTG study (Turner, 2014). In fact, multiple lines of investigation now strongly suggest not just that aggregate human activity is approaching ecological limits, but that it has already reached a state of significant *overshoot* beyond those limits. Overshoot is a qualitatively distinct regime for the design and operation of any adaptive or mitigating interventions (Catton, 1982). Effective societal responses to date have been significantly impaired by a lack of wide understanding of this harsh ecological reality. This gap in understanding facilitates the comforting — but erroneous — notion that it is prudent to delay difficult responses until after impacts are manifest. But delay is precisely one of the principle mechanisms that actually *causes* overshoot, and undermines the capability to damp the subsequent "crash".

Accordingly, it is suggested that there is now a clear need to develop what is here termed an *overshoot curriculum*, and to integrate this with formation of graduates in *all* disciplines. Further, a key pedagogical technique in delivering such a curriculum will be the systematic use of computational model ecologies. The Alife community is therefore uniquely positioned to contribute to this radical reform of higher education to meet what are, without exaggeration, the most profound challenges in the history of human civilization.

References

Catton, W. R. (1982). *Overshoot*. University of Illinois Press. https://goo.gl/mf4519

Forrester, J. W. (1982). *Principles of Systems*. MIT Press. https://goo.gl/BG7EUp

Gilding, P. (2012). *The Great Disruption*. Bloomsbury. https://goo.gl/6W1sRL

Meadows, D. H. et al. (1972). *The Limits to Growth*. Universe Books. http://goo.gl/XxgxpF

Rockström, J. et al. (2009). Planetary boundaries: exploring the safe operating space for humanity. *Ecology and Society*, 14(2): art. 32. http://goo.gl/A0Q6IB

Turner, G. (2014). Is global collapse imminent? Technical Report 4, Melbourne Sustainable Society Institute. http://goo.gl/Gxc7kD

Emergence of Competition between Different Dissipative Structures for the Same Free Energy Source

Stuart Bartlett[1,2], Seth Bullock[2]

[1]École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland
[2]University of Southampton, Southampton, SO17 1BJ, U.K.
stuart.bartlett@epfl.ch

Abstract

In this paper, we explore the emergence and direct interaction of two different types of dissipative structure in a single system: self-replicating chemical spot patterns and buoyancy-induced convection rolls. A new Lattice Boltzmann Model is developed, capable of simulating fluid flow, heat transport, and thermal chemical reactions, all within a simple, efficient framework. We report on a first set of simulations using this new model, wherein the Gray-Scott reaction diffusion system is embedded within a non isothermal fluid undergoing natural convection due to temperature gradients. The non-linear reaction which characterises the Gray-Scott system is given a temperature-dependent rate constant of the form of the Arrhenius equation. The enthalpy change (exothermic heat release or endothermic heat absorption) of the reaction can also be adjusted, allowing a direct coupling between the dynamics of the reaction and the thermal fluid flow.

The simulations show positive feedback effects when the reaction is exothermic, but an intriguing, competitive and unstable behaviour occurs when the reaction is sufficiently endothermic. In fact when convection plumes emerge and grow, the reaction diffusion spots immediately surround them, since they require a source of heat for the reaction to proceed. Then however, the proliferation of spot patterns dampens the local temperature, eventually eliminating the initial convection plume and reducing the ability of the spots to persist. This behaviour appears almost ecological, similar as it is, to competitive interactions between organisms competing for the same nutrient source.

Introduction

Spontaneous pattern formation is the primary exemplar of self-organisation in non-equilibrium systems, and remains a fascinating, enigmatic, and active field of research. Such phenomena has been observed and modelled in a wide range of systems including Dendritic Solidification (Ben-Jacob and Garik, 1990; Halsey, 2000; Vicsek, 1984), fluid convection (Pesch, 1996; Kadanoff, 2001), surfactant structures (Bachmann et al., 1992; Hanczyc and Szostak, 2004; Mayer et al., 1997; Ono, 2005), bacterial colony formation (Ben-Jacob, 1997, 1993), and reaction-diffusion systems (Lee et al., 1994; Turing, 1952; Pearson, 1993; Gray and Scott, 1985, 1994; Mahara et al., 2008; Virgo, 2011), see also Gollub and Langer (1999); Prigogine (1978); Kondepudi and Prigogine (2014). From an artificial life perspective, the dynamics of these (often very simple) systems are particularly poignant since they often exhibit properties analogous to those of living systems. We see behaviour such as self-replication, precariousness (Virgo, 2011), adaptation and response to external conditions, even primitive forms of evolution.

We are compelled towards the conclusion that the phenomenological separation between non-living and living systems might be bridged by a continuum of simpler, so-called 'dissipative structures' (we assume that conscious, intelligent life is the most complex example of all). We can relate some of the emergent patterns listed above to the origin of life on Earth, in particular the well-known self-replicating micelles, vesicles and droplets formed when surfactant molecules are dissolved in a polar solvent (Bachmann et al., 1992; Hanczyc and Szostak, 2004; Ono, 2005).

But we can also approach the problem on a more abstract level and pose the following question: Are the characteristic dynamics of life an inevitable, emergent property of non-equilibrium systems with a sufficient number of degrees of freedom and sufficient time? In this very general form, the answer is likely no, but it is still a worthy line of enquiry to try to delineate the types of systems, conditions, and timescales required for the emergence of life-like phenomena, in systems that are un-related to extant life. Thus in this work, we wished to explore whether non-living dissipative structures of different forms, might interact in ways reminiscent of biological systems.

The Gray-Scott reaction diffusion system (GSRDS) has been thoroughly investigated both theoretically (Gray and Scott, 1985, 1994; Mahara et al., 2008; Pearson, 1993; Virgo, 2011), and experimentally (Lee et al., 1994). One could even describe it as a descendant of the original ideas of morphogenetic pattern formation due to Turing (1952). The physical principles involved are simply the diffusion of two chemical species across a 2-dimensional domain, a non-linear reaction between the two species, and the addition and removal of the two species to and from the domain. Within certain regions of the parameter space of this model, the balance between supply of reactant, diffusion across concentration gradients, reaction into product, and removal of product, can produce a dizzying array of spontaneous patterns including self-replicating spots, stripes, waves, oscillations and spirals, among others. In this work we will be specifically interested in the self-replicating spot system, since we wish to explore the circumstances under which individual spots might compete not only with other spot patterns, but with completely different types of dissipative structure.

Alongside reaction diffusion phenomena, buoyancy-driven convection of a single phase fluid stands as a characteristic archetype of emergent non-equilibrium pattern formation. Such systems are defined by the presence of a sustained temperature gradient and associated differential heating. This induces a buoyancy force due to density differences, setting in motion a flow of the fluid which can feed back on itself, producing rising plumes. Eventually a steady state (in the statistical sense for those systems that are turbulent) is reached in which a pattern of rolls or more complex structures facilitates the delivery of heat from the warmer to the colder boundary. The resulting heat flux is far greater than that which would occur by diffusion alone (see e.g., Ahlers et al., 2009; Bartlett and Bullock, 2014; Johnston and Doering, 2009; Malkus, 1954; Manneville, 2006, and references therein). The various flow patterns which emerge in these convective systems constitute another class of dissipative structure; they form in the presence of a free energy gradient, and act to reduce that gradient. It appears that they are a manifestation of the system trying to return to equilibrium. The formation of dissipative structures does however appear to hinder the approach to equilibrium in some cases (Awazu and Kaneko, 2004), but a full discussion of this observation is beyond the scope of this paper.

Given the robust, pattern-forming properties of GSRDS's and fluid convection, as discussed above, one could naturally ask whether these two phenomena might interact in the same system under certain conditions. It is this question which motivated the present work. The majority of GSRDS models assume that the two chemical species are dissolved within a motionless fluid. However the system can readily be modified such that the solvent is able to exhibit a sustained flow.

Reactive flow systems are of immense industrial importance and have hence attracted significant interest (see e.g., Andres and Cardoso, 2012; Berenstein and Beta, 2011; Kee et al., 2005, and references therein). However, the idea of embedding the GSRDS within a moving fluid has been explored to a lesser extent. Ayodele et al. (2013) analysed the effects of imposing a pre-defined flow field upon a GSRDS. They performed a linear stability analysis on this new, more complex system and defined the conditions under which the system was no longer robust to finite perturbations. It was demonstrated that a simple flow produces uniform translation of the reaction diffusion patterns (as one would expect from Galilean invariance), but that differential advection can have a significant impact on the patterns formed by the chemical species concentrations.

While this paper achieved crucial milestones with regard to the modelling of reactive flow systems, it was only the chemical patterns which were an emergent result of the internal dynamics of the system. The flow fields were manually defined and imposed (they were boundary conditions). Similar systems have been modelled by other authors (Kreyssig and Dittrich, 2011). In the present work, we simulated systems in which both the chemical patterns *and* flow patterns emerged spontaneously. The primary achievement was the elucidation of conditions under which these two sets of phenomena directly interact with one another, in ways reminiscent of ecological phenomena.

In exploring this line of enquiry the principle difficulty was the construction of a simulation technique capable of representing the relevant physical processes, without being prohibitive in terms of computational power. Thus a new type of Lattice Boltzmann Model was derived and utilised (see Bartlett, 2014, for a comprehensive description of the method and its development).

In this paper we first present the basic elements of the model, in the form required for answering the questions posed above. We then illustrate the most interesting phenomena so far observed in such systems, in particular the emergence of a competitive dynamic between the two different types of dissipative structure in the system. We then conclude and make suggestions for the further exploration of this area.

The Chemical Lattice Boltzmann Model

Fluid dynamics

The LBM is a kinetic-based computational fluid dynamics method that can be used to numerically model a vast array of physical systems including single-phase (Chen and Doolen, 1998), multi-phase (Shan and Chen, 1993) and multi-component fluids (Arcidiacono et al., 2007), magnetohydrodynamic systems (Chen et al., 1991), oil-water-surfactant systems (Chen et al., 2000; Nekovee et al., 2000) and reactive flow systems (Frouzakis, 2011), (see Chen and Doolen, 1998, for a comprehensive review of the LBM). It derives from the non-equilibrium Boltzmann equation:

$$\frac{\partial f'}{\partial t} + \mathbf{v}\nabla f' = \Omega(f'), \tag{1}$$

which is the governing equation for the velocity distribution function $f'(\mathbf{x}, \mathbf{v}, t)$. This represents the mass of particles per unit volume moving at velocity $\mathbf{v}$, measured over a small volume element centred on position $\mathbf{x}$ at time t. Changes in f' within this volume element occur through advection by the particle motion (at velocity $\mathbf{v}$), and through collisions between particles within the element, encapsulated in the collision term $\Omega(f')$.

An appropriate discretisation and de-dimensionalisation leads to the evolution equation for the LBM on a discrete lattice:

$$f_i\left(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t\right)-f_i(\mathbf{x},t)=-\frac{1}{\tau_\nu}(f_i-f_i^{eq}) \quad (2)$$

where Δt is the time-step, τ_ν is the relaxation time and f_i^{eq} is the distribution of f at equilibrium. Use of this equation assumes that the approach to equilibrium is characterised by a single time-scale. It can be shown that the appropriate equilibrium distributions are the following (Wolf-Gladrow, 2000),

$$f_i^{eq}=\omega_i\rho\left[1+3\frac{\mathbf{e}_i\cdot\mathbf{u}}{c^2}+\frac{9}{2}\frac{(\mathbf{e}_i\cdot\mathbf{u})^2}{c^4}-\frac{3}{2}\frac{\mathbf{u}^2}{c^2}\right], \quad (3)$$

where ρ is the local fluid density, **u** is the local velocity, c is the lattice spacing and $\mathbf{e}_i$ are the discrete velocity vectors. In this work, the 2-dimensional D2Q9 model will be used, which utilises a square lattice with 8 velocities and rest particles. For this velocity set, the weights ω_i are $\omega_0 = 4/9$, $\omega_i = 1/9$ for $i = 1, 2, 3, 4$ and $\omega_i = 1/36$ for $i = 5, 6, 7, 8$. The velocity vectors are thus:

$$\begin{aligned}\mathbf{e}_0&=(0,0)\\ \mathbf{e}_{1,3},\mathbf{e}_{2,4}&=(\pm c,0),(0,\pm c)\\ \mathbf{e}_{5,6,7,8}&=(\pm c,\pm c)\end{aligned} \quad (4)$$

To calculate macroscopic fluid variables, the appropriate moments of the distribution functions must be taken:

$$\begin{aligned}\rho(\mathbf{x},t)&=\sum_i f_i(\mathbf{x},t)\\ \rho(\mathbf{x},t)\mathbf{u}(\mathbf{x},t)&=\sum_i \mathbf{e}_i f_i(\mathbf{x},t)\end{aligned} \quad (5)$$

Using the Chapman-Enskog expansion, it can be shown that at the macro-level, a fluid obeying the above equations satisfies the Navier-Stokes equations with a kinematic viscosity given by:

$$\nu=\frac{1}{3}\left(\tau_\nu-\frac{1}{2}\right)c^2. \quad (6)$$

The above equations are the main ingredients required to construct a LBM simulation for a single phase fluid. A large body of literature has demonstrated that this method can accurately simulate the behaviour of real fluids (see Chen and Doolen, 1998; Wolf-Gladrow, 2000, and references therein for further details concerning boundary conditions etc.).

Passive Scalars

As well as a single-phase fluid, the LBM can be modified to include extra components of various forms including internal energy (He et al., 1998; Peng et al., 2003; Shan, 1997) and passive scalars, such as chemical species (Ayodele et al., 2011, 2013; Arcidiacono et al., 2007). Let us first focus on the addition of the temperature field. All that is required is an extra distribution function:

$$g_i\left(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t\right)-g_i(\mathbf{x},t)=-\frac{1}{\tau_c}(g_i-g_i^{eq}) \quad (7)$$

where τ_c is the internal energy relaxation time and g_i^{eq} are the appropriate equilibrium distributions (He et al., 1998; Peng et al., 2003):

$$\begin{aligned}g_{i=0}^{eq}&=-\frac{2}{3}\rho\epsilon\frac{\mathbf{u}^2}{c^2}\\ g_{i=1,2,3,4}^{eq}&=\frac{1}{9}\rho\epsilon\left[\frac{3}{2}+\frac{3}{2}\frac{\mathbf{e}_i\cdot\mathbf{u}}{c^2}+\frac{9}{2}\frac{(\mathbf{e}_i\cdot\mathbf{u})^2}{c^4}-\frac{3}{2}\frac{\mathbf{u}^2}{c^2}\right]\\ g_{i=5,6,7,8}^{eq}&=\frac{1}{36}\rho\epsilon\left[3+6\frac{\mathbf{e}_i\cdot\mathbf{u}}{c^2}+\frac{9}{2}\frac{(\mathbf{e}_i\cdot\mathbf{u})^2}{c^4}-\frac{3}{2}\frac{\mathbf{u}^2}{c^2}\right].\end{aligned} \quad (8)$$

The internal energy density is represented by ϵ:

$$\begin{aligned}\rho(\mathbf{x},t)\epsilon(\mathbf{x},t)&=\rho(\mathbf{x},t)RT(\mathbf{x},t)\\ &=\sum_i g_i(\mathbf{x},t).\end{aligned} \quad (9)$$

The diffusivity of this extra component (the thermal diffusivity) is given by

$$\chi=\frac{2}{3}\left(\tau_c-\frac{1}{2}\right)c^2. \quad (10)$$

To implement the effects of buoyancy, a forcing term must be added to the fluid evolution equation, which then reads,

$$f_i\left(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t\right)-f_i(\mathbf{x},t)=-\frac{1}{\tau_\nu}(f_i-f_i^{eq})+F_i, \quad (11)$$

where

$$F_i=\frac{\mathbf{G}\cdot(\mathbf{e}_i-\mathbf{u})}{RT_0}f_i^{eq}, \quad (12)$$

R is the molar gas constant, T_0 is the mean temperature and $\mathbf{G}=\beta g_0(T-T_0)\hat{\jmath}$ is the vector representing the buoyant gravity force (β is the thermal expansion coefficient, g_0 is the acceleration of gravity and T is the local temperature).

With these modifications and appropriate boundary conditions (e.g., fixed temperature, fixed flux or zero flux), it is possible to simulate buoyancy driven convective systems. Again, a large body of work (see e.g., He et al., 1998; Peng et al., 2003; Shan, 1997) has validated the accuracy of this model (up to certain values of the Rayleigh number, above which, modifications are required).

In a similar manner to internal energy, other passive scalars can be added to the model. They have analogous evolution equations:

$$h_i^\sigma\left(\mathbf{x}+\mathbf{e}_i\Delta t, t+\Delta t\right)-h_i^\sigma(\mathbf{x},t)=-\frac{1}{\tau_\sigma}(h_i^\sigma-h_i^{eq,\sigma}) \quad (13)$$

where σ denotes the particular component, and τ_σ is the relaxation time for that component. The equilibrium distributions take on the familiar form:

$$h_{i=0}^{eq,\sigma} = -\frac{2}{3}\rho\psi_\sigma \frac{\mathbf{u}^2}{c^2}$$
$$h_{i=1,2,3,4}^{eq,\sigma} = \frac{1}{9}\rho\psi_\sigma \left[\frac{3}{2} + \frac{3}{2}\frac{\mathbf{e}_i \cdot \mathbf{u}}{c^2} + \frac{9}{2}\frac{(\mathbf{e}_i \cdot \mathbf{u})^2}{c^4} - \frac{3}{2}\frac{\mathbf{u}^2}{c^2}\right]$$
$$h_{i=5,6,7,8}^{eq,\sigma} = \frac{1}{36}\rho\psi_\sigma \left[3 + 6\frac{\mathbf{e}_i \cdot \mathbf{u}}{c^2} + \frac{9}{2}\frac{(\mathbf{e}_i \cdot \mathbf{u})^2}{c^4} - \frac{3}{2}\frac{\mathbf{u}^2}{c^2}\right]. \quad (14)$$

Reactions

Having described a thermal LBM which can simulate the advection and diffusion of passive scalars, let us now consider chemical reactions between a subset of those passive scalars. For convenience we will begin with an isothermal GSRDS. The equations of motion for the concentration fields of the two species are:

$$\frac{\partial \psi_A}{\partial t} = D_A \nabla^2 \psi_A - \psi_A \psi_B^2 + F(1 - \psi_A) \quad (15)$$
$$\frac{\partial \psi_B}{\partial t} = D_B \nabla^2 \psi_B + \psi_A \psi_B^2 - (F + R)\psi_B, \quad (16)$$

with the following autocatalytic reaction occurring: $A + 2B \rightarrow 3B$. The first terms on the RHS's represent diffusion, the second terms correspond to the non-linear reaction. The final terms in the two equations represent the inward or outward fluxes of A and B from or to external reservoirs. The inward flux of A is proportional to the feed rate F, and the local difference in concentration between ψ_A and unity. Similarly, the removal of B is proportional to $F + R$, and also to the local concentration difference between ψ_B and 0.

It is straightforward to implement these effects in the LBM. The distribution functions representing the mass of the two components evolve according to:

$$h_i^A(\mathbf{x} + \mathbf{e}_i\Delta t, t + \Delta t) - h_i^A(\mathbf{x}, t) = -\frac{1}{\tau_A}(h_i^A - h_i^{eq,A}) - \omega_i \psi_A \psi_B^2 + F(1 - \psi_A) \quad (17)$$
$$h_i^B(\mathbf{x} + \mathbf{e}_i\Delta t, t + \Delta t) - h_i^B(\mathbf{x}, t) = -\frac{1}{\tau_B}(h_i^B - h_i^{eq,B}) + \omega_i \psi_A \psi_B^2 - (F + R)\psi_B. \quad (18)$$

Although the field of reactive LBMs is less mature than that of simple fluid LBMs, modelling of GSRDSs has already been carried out and validated by Ayodele et al. (2011). Further testing was carried out in Bartlett (2014).

Enthalpy Changes

We know from the work of (Ayodele et al., 2013) that uniform advection of GSRDS systems produces a simple translation of the chemical patterns. We also know that differential advection can influence the linear stability of those structures. However in this work we aimed to uncover the dynamics of thermally driven systems, where both the fluid convection and chemical reactions are critically dependent on local temperature, and also influence the local temperature. In order to simulate these non-isothermal reactive flows, we require a temperature dependence within the reaction rate constant (this rate constant is set to unity for the standard GSRDS). We thus make use of the Arrhenius equation:

$$k(\mathbf{x}, t) = A e^{-E_a/T(\mathbf{x},t)} \quad (19)$$

where $k(\mathbf{x}, t)$ is the reaction rate at position $\mathbf{x}$ and time t, A is a parameter known as the frequency factor, E_a is the activation energy of the reaction, and T is the local temperature. Finally we must add an extra term to the internal energy evolution equation to account for the uptake and release of heat due to reactions:

$$g_i(\mathbf{x} + \mathbf{e}_i\Delta t, t + \Delta t) - g_i(\mathbf{x}, t) = -\frac{1}{\tau_c}(g_i - g_i^{eq}) - \omega_i k(\mathbf{x}, t)\psi_A \psi_B^2 \Delta H, \quad (20)$$

where ΔH is the enthalpy change of the reaction (positive for endothermic reactions and vice versa). With this thermal reactive LBM, we are in a position to simulate all the necessary phenomena: buoyancy-driven fluid flow, advection and diffusion of heat, advection and diffusion of chemical species, and thermal reactions between species.

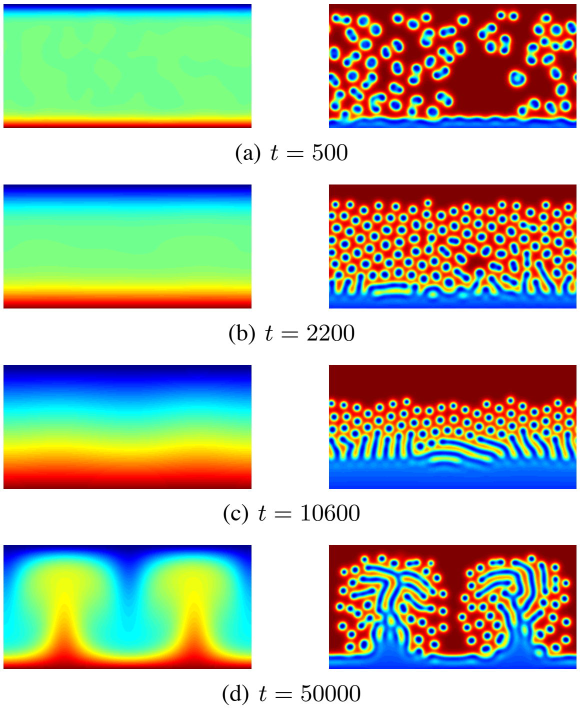
Competition for Free Energy Access

We now present an overview of the phenomena that can be exhibited by the above described model. For a more comprehensive exploration, please see Bartlett (2014).

First consider a system in which there is no uptake or release of heat during the reaction. Hence the presence of the chemical species does not influence the temperature field or fluid flow. The reaction rate is however influenced by the local temperature, and the chemical species are advected by the flow. The parameters used in these simulations are $E_a = 1.7$ and $A = e^{E_a/T_0}$, where T_0 is the mean temperature. The evolution of the system is shown in Figure 1.

We see that the convection of the fluid proceeds as normal (there is no feedback between the reaction and the fluid flow). The effects of the temperature dependent rate constant are also clearly visible. The temperature gradient between the warm lower boundary, cooler inner regions of the convection rolls, and cold upper boundary takes the chemical patterns from a steady state of pure B at the bottom of the system, through stripes and worms to self-replicating spots, all the way to a region of pure A near the top boundary. This ‘phase boundary layer’ is similar to the one visible in Figure 6.2 of (Bartlett, 2014), where it was caused instead by changes in the feed and removal rates, F and R.

We can now explore the impact of allowing a heat release from the autocatalytic reaction between the two

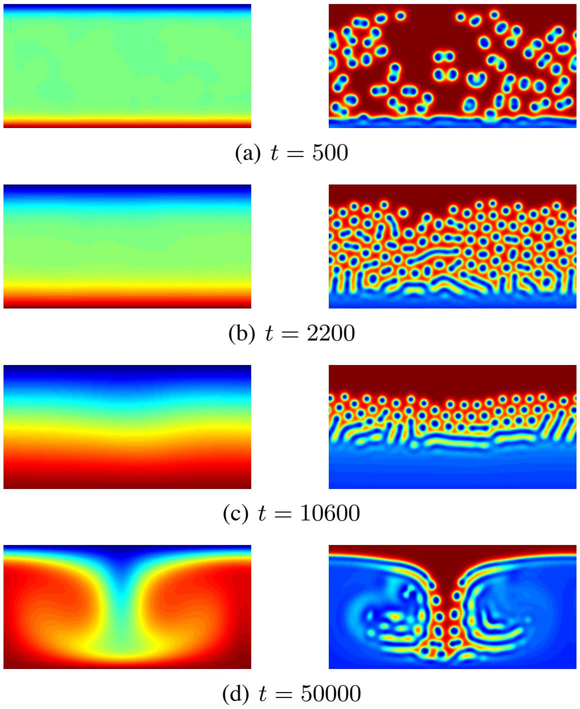

(a) $t = 500$

(b) $t = 2200$

(c) $t = 10600$

(d) $t = 50000$

Figure 1: Temperature (left column) and chemical order parameter $\phi = \psi_A - \psi_B$ (right column) fields for a thermally neutral ($\Delta H = 0$), convective GSRDS at several different times through the simulation.


(a) $t = 500$

(b) $t = 2200$

(c) $t = 10600$

(d) $t = 50000$

Figure 2: Temperature (left column) and chemical order parameter $\phi = \psi_A - \psi_B$ (right column) fields for an exothermic ($\Delta H = -1 \times 10^{-3}$), convective GSRDS at several different times through the simulation.

species. Figure 2 shows a strongly exothermic simulation with $\Delta H = -1 \times 10^{-3}$.

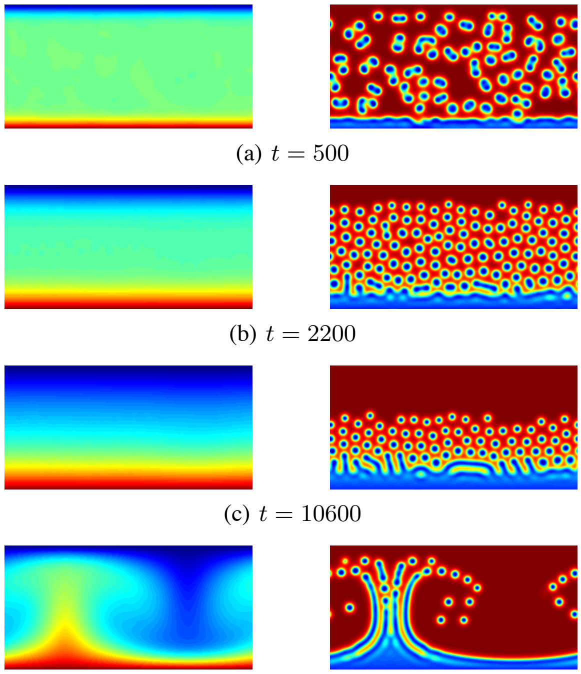
In this case we see a dominating positive feedback effect where the heat released from the reaction enhances the reaction rate further, releasing even more heat. This dynamic continues indefinitely and the entire system becomes swamped by the B substance and its temperature continues to increase.

Finally, let us move to the endothermic case. Figure 3 shows a simulation in which $\Delta H = 2 \times 10^{-3}$. We see that the convection pattern is still able to form (note that with an aspect ratio of 2, the fluid-only system sometimes forms two convection rolls, and sometimes four, as in Figure 1, therefore the difference in the number of rolls between these two simulations is not related to the chemical phenomena). What is also visible is that the reaction-diffusion 'phase boundary layer', seen in Figure 1, is present, but it seems to be slightly more confined to the lower end of the domain, and the proliferation of spot patterns is subdued in this endothermic situation. This is a result of the uptake of heat by the reaction, which forces the system to try to extract more heat from the lower boundary to compensate.

Now that we have observed an interaction between the two different types of dissipative structure (the reaction diffusion patterns affecting the heat flux, which itself is enhanced by the convection rolls), we can consider whether it is possible for the effects of one to completely diminish the other. We could for example, make the reaction even more endothermic.

When this is implemented, the reaction diffusion patterns end up confined to the lower boundary, the only place where the temperature can be maintained sufficiently high. If however, we reduce the activation energy of the reaction to $E_a = 0.2$, and further increase the reaction enthalpy to $\Delta H = 25 \times 10^{-3}$, a fascinating, oscillatory dynamic sets in. This is best viewed through the animation of the simulation: Bartlett (2015).

Initially, spots form across the domain, however they soon cool their environment so much that they can no longer persist. In fact the temperature begins to dip below even that of the cold upper boundary, and hence a small number of spots survive there. There is thus a large temperature gradient between the lower boundary, maintained at a high temperature, and the inner region of the domain. This gradient provides a strong driving force for the formation of convective plumes (the seeds of convection rolls), which sometimes burst into existence. However when this happens, there is a rapid proliferation of chemical activity, as the spot patterns appear to move to this new found heat source. The replication process removes heat from that locality, so much in fact that the



(a) $t = 500$

(b) $t = 2200$

(c) $t = 10600$

(d) $t = 50000$

Figure 3: Temperature (left column) and chemical order parameter $\phi = \psi_A - \psi_B$ (right column) fields for an endothermic ($\Delta H = 2 \times 10^{-3}$), convective GSRDS at several different times through the simulation.

initial convection plume becomes completely damped out. This competitive dynamic continues indefinitely, with the sporadic appearance of convection plumes, which are immediately leapt upon by ‘clouds’ of spot structures, the action of which leads to their own demise, and the demise of the plume.

In order to illustrate this interaction quantitatively, Figure 4 displays the time series of the vertical convective heat flux $Q_{y_c} = \overline{u_y T}$ (left axis and blue dashed line) and the spot pattern population N_{RDS} (right axis and green solid line). It is difficult to numerically measure the proliferation of convection rolls, but the vertical convective heat flux at least illustrates changes in the degree of correlation between the temperature field and the vertical velocity field. It is this correlation which indicates the presence of spatially ordered heat movements. Tracking the population of reaction diffusion spots is somewhat more straightforward, given that they are discrete structures (apart from during their transient, division phase).

The two signals in Figure 4 are intermittent, but we see that there is a clear connection between them: they appear to be anti-correlated. The oscillations of both measures also seem to have a fairly consistent frequency. As expected from the animation of the simulation, most peaks in heat flux are followed by a peak in spot population. That peak then coincides with a reduction in heat flux. Indeed the increased growth of the spots causes a decline in heat flux. A thorough statistical analysis of this and other time series from the same ensemble is currently being carried out.

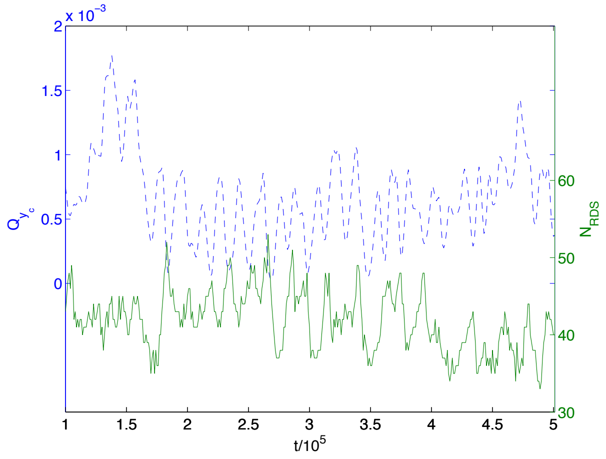


Figure 4: Population dynamics of the spot patterns within a thermal convective GSRDS. The left axis and blue dashed line corresponds to the vertical convective heat flux whereas the right axis and green solid line follows the number of individual spot patterns present in the system.

Conclusions

We have presented a new LBM capable of simulating an immense spectrum of physical phenomena. The full scope of the model can only be touched upon here. In this work, we have used it to explore the interactions between different types of dissipative structure, forming in a single system. The model is not a hybrid of different types, and does not require vast amounts of computational power. Its construction is also simple, intuitive, and physically-based. As shown by other authors previously, the isothermal chemical LBM can readily simulate simple chemical systems including the GSRDS.

What is novel about the work presented here is that we have relaxed the isothermal assumption and introduced a temperature dependent rate constant based on the Arrhenius equation. As far as we are aware, this is the first time that a thermal GSRDS has been simulated within a convective fluid.

When the reaction does not release or absorb heat, the convection pattern forms as normal, and the chemical species exhibit a range of patterns across the spatial temperature gradient, due to the modulation of the rate constant with temperature. An exothermic scheme causes the reactive phase boundary to expand as it positively feeds back upon itself. With even more negative values of the enthalpy change, the positive feedback effect becomes so large that the system is overwhelmed by the single chemical species state.

Conversely, an endothermic reaction has a damping effect on both the proliferation of chemical structures and convection patterns. If the reaction is very endothermic then an oscillatory behaviour emerges, in which neither spot structures nor convection rolls can stably persist. Instead, any rising convective plume becomes rapidly surrounded by spot patterns, as the additional delivery of heat from the plume is taken up by the action of the endothermic spots.

What is remarkable about this behaviour is that it is reminiscent of an ecological scenario, in which two species, both competing for the same food source, deplete that source so quickly that it can no longer provide for either of them. Indeed a reduction in the enthalpy of reaction allows both spots and convection rolls to form. But when ΔH was sufficiently large, the spot patterns deplete a necessary component of their existence (a heat source) so fast that they destroy the conditions under which they can survive.

These results are suggestive of the possibility that ecological dynamics are more fundamental and older than biological phenomena. One can imagine other-worldly scenarios in which thermal or chemical gradients thrust elaborate patterns into existence, acting to reduce those gradients but being sustained by them. It seems likely that several such structures could coexist and interact. To take an example close to those in this paper, imagine if there were large gradients of chemical potential within the gases comprising Jupiter's red spot, and non-linear reactions between the relevant chemical species.

Further Work

The next chapter of this story should be to explore even more complex emergent ecologies in these reactive flow systems. Can these thermal spot patterns exhibit precariousness, the property of being able to emerge in a certain environment, and then go on to colonise regions that were previously inhospitable (Virgo, 2011)? An initial exploration of this question is presented in Bartlett (2014). Conditions under which the spots were shown to be precarious were not found. However, given the intimidating size and dimensionality of the parameter space, it is difficult to rule out any possibilities.

Might it be possible to observe strong hysteresis and some form of primitive evolution and adaptation of these structures? Ultimately, can this modelling paradigm help us uncover systems spanning the phenomenological gulf between non-living and living dissipative structures? Work to investigate these mysteries is currently in progress.

One could also increase the size of the reaction diffusion spots by tuning the diffusion coefficients of the chemical species. The structures could be made to be the same length scale as the convection rolls. This would undoubtedly modify the dynamics already presented in this paper.

Another major direction to pursue is systems in which there is only one free energy source. Removing the feed and removal processes from the GSRDSs would achieve this, but then the characteristic patterns would not form. A materially closed system with only a temperature gradient across it was simulated in Bartlett (2014). There were two chemical species present with a simple linear reaction transforming one to the other. It was found that the presence of the chemical species enhanced the ability of the system to deliver heat from one boundary to the other. This enhancement was proportional to the total mass of chemical species dissolved in the solvent fluid. The next phase will be to employ more complex, nonlinear reaction schema such that individuated structures might appear.

Acknowledgements

This work was supported by an EPSRC Doctoral Training Centre grant (EP/G03690X/1). We would also like to thank three anonymous reviewers for their constructive and insightful comments.

References

Ahlers, G., Grossmann, S., and Lohse, D. (2009). Heat transfer and large scale dynamics in turbulent rayleigh-bénard convection. *Rev. Mod. Phys.*, 81:503–537.

Andres, J. T. H. and Cardoso, S. S. S. (2012). Convection and reaction in a diffusive boundary layer in a porous medium: Nonlinear dynamics. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(3).

Arcidiacono, S., Karlin, I. V., Mantzaras, J., and Frouzakis, C. E. (2007). Lattice boltzmann model for the simulation of multicomponent mixtures. *Phys. Rev. E*, 76:046703.

Awazu, A. and Kaneko, K. (2004). Relaxation to equilibrium can be hindered by transient dissipative structures. *Phys. Rev. Lett.*, 92:258302.

Ayodele, S., Raabe, D., and Varnik, F. (2013). Lattice boltzmann modeling of advection-diffusion-reaction equations: Pattern formation under uniform differential advection. *Communications in Computational Physics*, 13:741–756.

Ayodele, S. G., Varnik, F., and Raabe, D. (2011). Lattice boltzmann study of pattern formation in reaction-diffusion systems. *Phys. Rev. E*, 83:016702.

Bachmann, P. A., Luisi, P. L., and Lang, J. (1992). Autocatalytic self-replicating micelles as models for prebiotic structures. *Nature*.

Bartlett, S. (2014). *Why is life? An assessment of the thermodynamic properties of dissipative, pattern-forming systems*. PhD thesis, University of Southampton.

Bartlett, S. (2015). Reaction-diffusion-convection simulation with strongly endothermic reaction. `https://www.youtube.com/watch?v=mIfN6h-uRqE`.

Bartlett, S. and Bullock, S. (2014). Natural convection of a two-dimensional boussinesq fluid does not maximize entropy production. *Phys. Rev. E*, 90:023014.

Ben-Jacob, E. (1993). From snowflake formation to growth of bacterial colonies. *Contemporary Physics*, 34(5):247–273.

Ben-Jacob, E. (1997). From snowflake formation to growth of bacterial colonies ii: Cooperative formation of complex colonial patterns. *Contemporary Physics*, 38(3):205–241.

Ben-Jacob, E. and Garik, P. (1990). The formation of patterns in non-equilibrium growth. *Nature*, 343:523–530.

Berenstein, I. and Beta, C. (2011). Flow-induced control of chemical turbulence. *The Journal of Chemical Physics*, 135(16).

Chen, H., Boghosian, B. M., Coveney, P. V., and Nekovee, M. (2000). A ternary lattice boltzmann model for amphiphilic fluids. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 456(2000):2043–2057.

Chen, S., Chen, H., Martnez, D., and Matthaeus, W. (1991). Lattice boltzmann model for simulation of magnetohydrodynamics. *Phys. Rev. Lett.*, 67:3776–3779.

Chen, S. and Doolen, G. D. (1998). Lattice boltzmann method for fluid flows. *Annual Review of Fluid Mechanics*, 30(1):329–364.

Frouzakis, C. (2011). Lattice boltzmann methods for reactive and other flows. In Echekki, T. and Mastorakos, E., editors, *Turbulent Combustion Modeling*, volume 95 of *Fluid Mechanics and Its Applications*, pages 461–486. Springer Netherlands.

Gollub, J. P. and Langer, J. S. (1999). Pattern formation in nonequilibrium physics. *Rev. Mod. Phys.*, 71:S396–S403.

Gray, P. and Scott, S. (1994). *Chemical Oscillations and Instabilities: Non-linear Chemical Kinetics*. International Series of Monographs on Chemistry. Clarendon Press.

Gray, P. and Scott, S. K. (1985). Sustained oscillations and other exotic patterns of behavior in isothermal reactions. *The Journal of Physical Chemistry*, 89(1):22–32.

Halsey, T. C. (2000). Diffusion-limited aggregation: a model for pattern formation. *Physics Today*, 53(11):36–41.

Hanczyc, M. M. and Szostak, J. W. (2004). Replicating vesicles as models of primitive cell growth and division. *Current opinion in chemical biology*, 8(6):660–664.

He, X., Chen, S., and Doolen, G. D. (1998). A novel thermal model for the lattice boltzmann method in incompressible limit. *Journal of Computational Physics*, 146(1):282 – 300.

Johnston, H. and Doering, C. R. (2009). Comparison of turbulent thermal convection between conditions of constant temperature and constant flux. *Phys. Rev. Lett.*, 102:064501.

Kadanoff, L. P. (2001). Turbulent heat flow: Structures and scaling. *Physics Today*, 54(8):34–39.

Kee, R., Coltrin, M., and Glarborg, P. (2005). *Chemically Reacting Flow: Theory and Practice*. Wiley.

Kondepudi, D. and Prigogine, I. (2014). *Modern Thermodynamics: From Heat Engines to Dissipative Structures*. Wiley.

Kreyssig, P. and Dittrich, P. (2011). Reaction flow artificial chemistries. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems, Paris*, pages 431–437. MIT Press.

Lee, K.-J., McCormick, W. D., Pearson, J. E., and Swinney, H. L. (1994). Experimental observation of self-replicating spots in a reaction-diffusion system. *Nature*, 369(6477):215–218.

Mahara, H., Suzuki, K., Jahan, R. A., and Yamaguchi, T. (2008). Coexisting stable patterns in a reaction-diffusion system with reversible gray-scott dynamics. *Phys. Rev. E*, 78:066210.

Malkus, W. V. (1954). The heat transport and spectrum of thermal turbulence. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 225(1161):196–212.

Manneville, P. (2006). Rayleigh-bénard convection: Thirty years of experimental, theoretical, and modeling work. In Mutabazi, I., Wesfreid, J. E., and Guyon, E., editors, *Dynamics of Spatio-Temporal Cellular Structures*, volume 207 of *Springer Tracts in Modern Physics*, pages 41–65. Springer New York.

Mayer, B., Köhler, G., and Rasmussen, S. (1997). Simulation and dynamics of entropy-driven, molecular self-assembly processes. *Phys. Rev. E*, 55:4489–4499.

Nekovee, M., Coveney, P. V., Chen, H., and Boghosian, B. M. (2000). Lattice-boltzmann model for interacting amphiphilic fluids. *Phys. Rev. E*, 62:8282–8294.

Ono, N. (2005). Computational studies on conditions of the emergence of autopoietic protocells. *BioSystems*, 81(3):223–233.

Pearson, J. E. (1993). Complex patterns in a simple system. *Science*, 261(5118):189–192.

Peng, Y., Shu, C., and Chew, Y. T. (2003). Simplified thermal lattice boltzmann model for incompressible thermal flows. *Phys. Rev. E*, 68:026701.

Pesch, W. (1996). Complex spatiotemporal convection patterns. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 6(3):348–357.

Prigogine, I. (1978). Time, structure, and fluctuations. *Science*, 201(4358):777–785.

Shan, X. (1997). Simulation of rayleigh-bénard convection using a lattice boltzmann method. *Phys. Rev. E*, 55:2780–2788.

Shan, X. and Chen, H. (1993). Lattice boltzmann model for simulating flows with multiple phases and components. *Phys. Rev. E*, 47:1815–1819.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 237(641):37–72.

Vicsek, T. (1984). Pattern formation in diffusion-limited aggregation. *Phys. Rev. Lett.*, 53:2281–2284.

Virgo, N. D. (2011). *Thermodynamics and the structure of living systems*. PhD thesis, University of Sussex.

Wolf-Gladrow, D. (2000). *Lattice-Gas Cellular Automata and Lattice Boltzmann Models: An Introduction*. Number no. 1725 in Lattice-gas Cellular Automata and Lattice Boltzmann Models: An Introduction. Springer.

Planarity as a driver of Spatial Network structure

Garvin Haslett and Markus Brede

Institute for Complex Systems Simulation, School of Electronics and Computer Science,
University of Southampton, Southampton SO17 1BJ, United Kingdom
g.a.haslett@soton.ac.uk

Abstract

In this paper we introduce a new model of spatial network growth in which nodes are placed at randomly selected locations in space over time, forming new connections to old nodes subject to the constraint that edges do not cross. The resulting network has a power law degree distribution, high clustering and the small world property. We argue that these characteristics are a consequence of two features of our mechanism, growth and planarity conservation. We further propose that our model can be understood as a variant of random Apollonian growth. We then investigate the robustness of our findings by relaxing the planarity. Specifically, we allow edges to cross with a defined probability. Varying this probability demonstrates a smooth transition from a power law to an exponential degree distribution.

Introduction

In the last decade and a half tools from network science have been increasingly used to investigate a wide range of complex systems. Work within the discipline has sought to elucidate network structures not traditionally considered within the existing mathematical literature (Barabási and Albert, 1999; Watts and Strogatz, 1998; Brede, 2011) which are typically characterised by the power law degree distribution, high clustering and the small world property. Concomitantly systems as diverse as the protein network (Jeong et al., 2001), the Internet (Faloutsos et al., 1999), academic authorship (Newman, 2001) and the C. elegans neuronal network (Amaral et al., 2000; Haruna, 2011) have been examined to ascertain the extent to which they exhibit these properties. The insights gained from such studies benefit Alife practitioners in that they inform more nuanced representations of population structure in simulation (Buesser and Tomassini, 2012; Salathé et al., 2010; Lever et al., 2014).

More recently the sub-discipline of spatial networks has emerged as a subject in its own right (Barthélemy, 2011). The distinguishing feature here is that the nodes are ascribed a position in $\mathbb{R}^2$ and the distance between them are described, typically, by the Euclidean metric. The chief aim being to investigate the extent to which constraining connectivity in a manner related to node proximity influences system organisation. Application domains for these techniques include city science (Cardillo et al., 2006; Xie and Levinson, 2007; Jiang, 2007; Barthélemy and Flammini, 2008; Masucci et al., 2009; Chan et al., 2011; Courtat et al., 2011; Strano et al., 2012; Levinson, 2012; Rui et al., 2013; Gudmundsson and Mohajeri, 2013), electronic circuits (i Cancho et al., 2001; Bassett et al., 2010; Miralles et al., 2010; Tan et al., 2014), wireless networks (Huson and Sen, 1995; Lotker and Peleg, 2010), leaf venation (Corson, 2010; Katifori et al., 2010), navigability (Kleinberg, 2000; Lee and Holme, 2012; Huang et al., 2014) and transportation (Gastner and Newman, 2006; Louf et al., 2013).

In common with the parent discipline, a key aim of the spatial network literature is to establish appropriate null models for use as reference cases. A significant example amongst these is the random geometric graph which has proved useful to physicists working within continuum percolation (Balberg, 1985; Quintanilla et al., 2000) and has been the subject of extensive mathematical investigations (Dall and Christensen, 2002). A brief description of this model is that nodes are placed at random on the surface of a torus and pairs that lie within a specified maximum distance of each other are connected. Significantly, a power law degree distribution can only be observed in this model when the nodes themselves are are distributed inhomogeneously in space (Herrmann et al., 2003; Barnett et al., 2007; Bullock et al., 2010).

Other attempts have been made to discover spatial mechanisms which give rise to scale free degree distributions. A key review in this regard is that of Hayashi (Hayashi, 2006) which outlined three mechanisms: link length penalisation, space filling and embedding a scale-free network on a lattice. The first of these classes constrains the creation of a new edge based on some function of its length. The random geometric graph just mentioned is the fundamental example of these. Subsequent adaptions that demonstrate the power law property create connections based on a product of node weight and a power of the distance (Masuda et al., 2005; Manna and Sen, 2002).

The second class recursively partitions the space by adding new nodes to the plane and then connects them to the existing graph. One approach places the nodes randomly, selects the nearest edge and connect to one of its end nodes (Mukherjee and Manna, 2006). The authors posit that the power law ensues due to equivalence, in the limit of infinite system size, to a seminal model by Dorogovetsev et al. (Dorogovtsev et al., 2001). The second example in this class, Apollonian Networks (Andrade Jr et al., 2005; Doye and Massen, 2004), are a fractal construction that begin with a triangle in the plane which is then trisected to produce K_4. This step is repeated ad infinitum with all triangles present at each stage being further trisected.

The final class assigns an implicit degree k to all the nodes in a lattice. Members of the lattice are then selected at random and connected to their k nearest neighbours subject to a distance constraint (Ben-Avraham et al., 2003; Rozenfeld et al., 2002).

In the next section we present a model, named planar growth, which is a contribution to the space filling class. The model is so named because it is a growing process that only accepts new edges that do not cross each other in $\mathbb{R}^2$. Alongside it we also describe two models, no growth and no planarity that are are used as reference cases to illuminate the outcomes observed in the planar growth model. In the subsequent section we present our initial results from these models chief among them that a power law degree distribution is observed. We then produce a variation of the Random Apollonian Network which places the nodes uniformly in space. This new model is identical to planar growth while making clear a relationship with existing Apollonian literature. It also lends itself to more efficient realisation and we use it to produce networks of a greater scale. In the penultimate section we look at the consequences of relaxing the planarity constraint. In the final section we present our conclusions.

Models

Planar growth (PG) takes place upon a unit Euclidean square with rigid boundary conditions. We wish to begin the process with a small, i.e. statistically insignificant, population that has average degree m, the number of new connections that will be made for each new node when we begin the growth process proper. To do so we place ten nodes uniformly at random within the square. Each new node is connected to one of those already present; this target node being chosen so that the new edge does not cross an existing one.

Once all ten nodes have been placed, unconnected pairs are chosen randomly and edges added between them, again subject to the caveat that planarity is maintained. We continue until either $m \times 10$ edges are present or all possible pairs have been tried.

The algorithm now enters the growing phase where the following steps are repeated n times, resulting in a network of size $n + 10$ nodes and $m(n + 10)$ edges:

Step 1
Place a new node uniformly at random within the square.

Step 2
Choose an existing node at random. If all existing nodes have been tried then reject the current node and return to step 1.

Step 3
Connect the new node and the existing node. If this connection results in crossed edges then reject it and repeat step 2, otherwise accept it and proceed.

Step 4
If the new node now has m connections continue to step 5. Otherwise attempt to acquire another connection for the new node by returning to step 2.

Step 5
If n nodes have been added to the network then finish, otherwise return to step 1.

The algorithm just outlined consists of two ingredients: (i) growth and (ii) planarity conservation. In the following section we make use of two further models, each containing just one of these two aspects. Then, in a manner following Barabási & Albert (Barabási and Albert, 1999), we compare outcomes between the original and these simpler models thereby determining the extent to which growth and planarity contribute to the overall outcome.

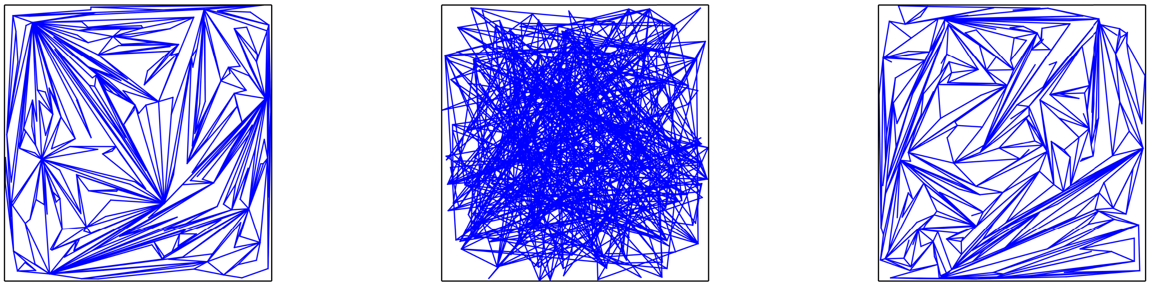
The first of these, the no planarity model, is similar to PG in that it is a growth process on a unit square. It differs in that edge connections are always allowed. This scenario is indistinguishable from the uniform attachment model originally introduced by Barabási & Albert which was shown to have an exponential degree distribution (Barabási et al., 1999).

The other, no growth, places n nodes at random on the unit square. Pairs of these nodes are then chosen at random and connections formed between them. Planarity conservation is still enforced meaning that new edges are only permitted where they do not cross existing ones. This algorithm continues until $n \times m$ edges have been added to the network.

Results

In figure 1 we present three visualisations of small ($n = 250$) graphs produced by each of the mechanisms. Qualitatively, the no planar plot has the most unstructured appearance. By contrast the PG and the no growth scenarios both exhibit open regions that are not crossed by edges and a 'bunching' effect whereby edges connected to higher degree hubs are spread out in a fan like structure. It also appears that the hubs for the planar growth experiment are of a higher degree than those of the no growth model.

We investigate the node degree quantitatively by plotting the cumulative degree distribution of two PG experiments, of orders $n = 10^3$ and $n = 10^4$ and $m = 2$, alongside a no planarity experiment of order $n = 10^4$ and a no growth



(a) Planar growth

(b) No planar

(c) No growth

Figure 1: Visualisation of planar growth, no planar and no growth networks with $n = 250$ and $m = 2$.

experiment of order $n = 5 \times 10^3$ in figure 2a. Each point is the average of the degree data from twenty experiments. Note that the restricted size of the no growth case is due to computational limits. These results show the degree distributions of both reference cases to be exponential. This is in contrast to the two planar growth cases; both of which seem to approximate a power law distribution, i.e. the probability, p_k of observing degree k is distributed as $p_k \sim k^{-\alpha}$. We have estimated the exponent in the $n = 10^4$, $m = 2$ case using the method of Maximum Likelihood Estimators outlined in Clauset et al. (Clauset et al., 2009) finding it to be $\alpha_{m=2} = 2.84 \pm 0.01$.

We present further statistical evidence for the presence of power laws in the form of scaling of the maximum degree with system size for different network types. In figure 2b we plot how the mean maximum degree observed during these experiments varies with the size of the network. Following Newman (Newman, 2003) we also plot $\langle k_{max} \rangle \sim n^{1/(\alpha-1)}$, where $\langle k_{max} \rangle$ is the mean value of the maximum degree. This relationship is the analytically calculated value of $\langle k_{max} \rangle$ under the assumption that k is distributed as a power law with exponent α. Specifically, we have used the exponent $\alpha_{m=2}$ estimated above and figure 2b demonstrates good agreement with the observations.

By standard techniques it can be shown that the expected value of the ith member of a sequential ordering of the random variables of an exponential distribution with parameter λ is $E[X_i] = H_i/\lambda$ where H_i is the ith harmonic number. We therefore expect, for a network with an exponential degree distribution, $\langle k_{max} \rangle \sim H_n$, i.e. $\langle k_{max} \rangle$ will grow logarithmically. The plots for the No Planarity and No Growth networks both conform to this prediction.

Taken together these plots indicate that the planar growth networks match expected behaviour for a power law degree distribution while, conversely, the two reference cases match that of an exponential one. However, note that in the planar growth cases the cumulative plot only extends over one order of magnitude along the k axis, an outcome that needs confirmation for larger system sizes. We will address this matter in the following section. Furthermore, the fact that the no growth and the no planarity models result in an exponential distribution leads us to conclude that neither growth nor planarity is sufficient to account for the scale free distribution; rather it is the combination of the two that is required. It is also interesting to note that although the no growth appeared visually similar to PG it is, in this regard, more similar to no planarity.

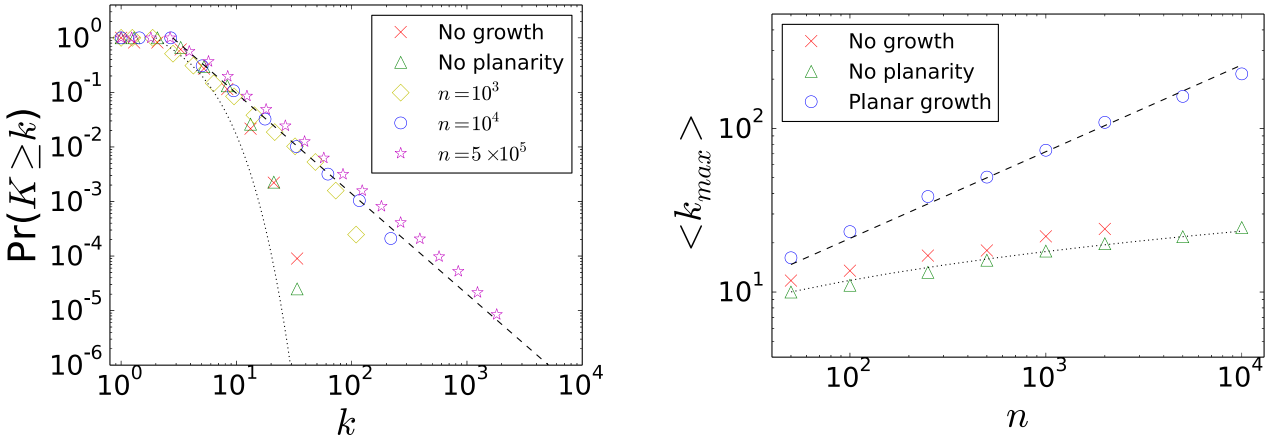
We now determine whether planar growth gives rise to the small-world property. Informally, small world networks have a high degree of community structure while maintaining a low average shortest path length. As such, they are believed to underpin the efficient transfer of information in large populations originally considered by Milgram (Milgram, 1967).

Community structure can be quantified by first introducing the clustering for an individual node i:

$$c_i = \frac{2t_i}{k_i(k_i - 1)}$$

where t_i is the number of triangles which have i as a vertex and k_i is the degree of node i. Therefore, $c_i = 0$ when none of i's neighbours are connected and $c_i = 1$ when they all are. Clustering c for a whole network is then the average of c_i over all the nodes in the network. Small worlds are then those networks that display high c and a logarithmic increase in l, the average number of hops in the shortest path between pairs of nodes, with system size n (Watts and Strogatz, 1998).

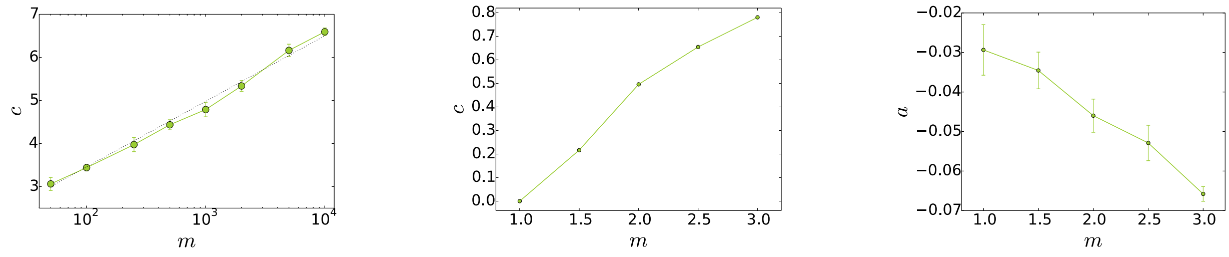
In a random network of size n and average degree m clustering follows $c_{\text{random}} \sim m/n$ and average shortest path length scales as $l_{\text{random}} \sim \ln(n)/\ln(m)$ (Watts and Strogatz, 1998); so for an $n = 10^4$, $m = 2$ random network this gives estimates of $c_{\text{random}} = 0.0002$ and $l_{\text{random}} = 13.29$. By comparison the observed values for a PG network using these values of n and m are $c_{m=2} = 0.492 \pm 0.002$ and $l_{m=2} = 6.6 \pm 0.1$, giving a first indication that planar net-



(a) Cumulative degree density

(b) Average maximum degree

Figure 2: Cumulative degree density for networks created using PG of orders $n = 10^3$ and 10^4, APG of order 5×10^5, No Planarity of order $n = 10^4$ and No Growth of order $n = 5 \times 10^3$. Dashed line is the best fit of APG experiment, a power law with exponent, $\alpha_{\text{APG}} = 2.84$. Dotted line is the degree distribution predicted by Barabási et al. (Barabási et al., 1999). (2a). Average maximum degree observed for PG with $m = 2$, No Planarity and No Growth networks of varying order. Error bars representing standard deviation not shown but are similar in magnitude to the points. Dashed line is the expected value of the maximum degree for a power law with exponent $\alpha_{m=2} = 2.84$, the estimated value of the exponent in the $n = 10^4, m = 2$ case. Dotted line is a plot of a logarithmic function. (2b).



(a) Path length varying with n.

(b) Clustering varying with m.

(c) Assortativity varying with m

Figure 3: Average shortest path length for $m = 2$ networks with varying size. For each network size twenty experiments were performed. Dashed line is a logarithmic function. (3a). Average clustering observed in twenty PG networks with $n = 10^4$ and varying m (3b). Average assortativity observed in twenty PG networks with $n = 10^4$ and varying m. (3c)

work growth indeed results in a small world. We further investigate the scaling of path length with system size in figure 3a where we plot l against n for a series of networks with $m = 2$. The straight line on this plot indicates that l is growing logarithmically with n confirming the small world claim.

Figure 3b plots how the clustering coefficient, c, varies with m for networks of a constant size. What we see is clustering rising from $c_{m=1} = 0.0$ to $c_{m=3} = 0.7806 \pm 0.0003$. These high values of c are the consequence of new nodes being placed within a v-shape formed by three nodes and two edges. When this happens formation of a new triangle becomes likely since the edges constrain the possible connections the new node can make, in particular increasing the likelihood that it will connect to the apex of the v shape and one of its neighbours. It is interesting to note at this point that the clustering for the no growth experiment is $c_{NG} = 0.29 \pm 0.02$ while that for a $n = 5000$, $m = 2$ network is $c_{n=5000} = 0.497 \pm 0.006$ indicating that similar tendencies exist in the no growth case but not to the extent that they act in PG.

Finally for this section figure 3c plots how the assortativity varies with m for PG networks of order $n = 10^4$, showing it to decrease gradually from $a_{m=1} = -0.029 \pm 0.006$ to $a_{m=3} = -0.066 \pm 0.002$, i.e. the networks are mildly disassortative and this tendency increases as m increases. These results are in line with the well known fact that random scale-free networks are disassortative (Maslov et al., 2004; Park and Newman, 2003). A partial explanation of this phenomenon that has been offered is that there is a limited amount of possible edges that can lie between high degree hubs (Maslov et al., 2004). So, in general, a scale-free network must feature connections between high and low degree nodes.

Apollonian Planar Growth

We now address the issue of the small system sizes in the previous section. The computational problem is that repeatedly checking the existing network for edge crossings is time consuming so we therefore seek an algorithm that realises PG more efficiently. To do so, we modify an existing process of network growth; the Random Apollonian network (RAN) (Zhou et al., 2005). This construction begins with K_4 embedded on the plane and one of its three faces is chosen at random. A node is then placed within this face and connected to the three vertices, thereby replacing the existing face with three new ones. The process continues, choosing a face of the triangulation uniformly at random and trisecting it, ad infinitum.

The original RAN did not ascribe an explicit position to its nodes. This leads to an interpretation that each new node lies in the barycentre of its containing triangle. Furthermore, note that there is a similarity with preferential attachment here in that the likelihood of a node being selected to receive a new connection is directly proportional to the number of faces it is a vertex of; a quantity that is equal to the number of edges incident to the node. Two points should be made here, firstly, there is a densification effect as new nodes a more likely to form in the vicinity of existing ones. Secondly, the degree distribution can be calculated analytically and is a power law with exponent equal to 3.

We modify the RAN to place nodes uniformly by adding two features. Firstly, we weight face selection by area, i.e. a face of area 0.5 units is twice as likely to be selected as one of 0.25 units. Secondly, when a new node is created, it is placed uniformly at random within the chosen face. This model is exactly planar growth on a triangle using degree of connection $m = 3$ and in light of this fact we name it Apollonian planar growth (APG). Computationally, the enforced trisecting may be represented efficiently as a growing ternary tree which, in turn, can be exploited to grow networks that are an order of magnitude larger than those created with PG.

Returning to figure 2a we see this increase in size reflected in the plot of APG's degree distribution which extends over two orders of magnitude along the k axis. This evidence, in combination with the fact that APG is a variation upon a process known to produce a scale free network leads us to conclude that PG itself results in a power law degree distribution.

Planarity Relaxation

The distinct differences between the PG and no planarity degree distributions lead us to consider how robust our process is. To investigate this we introduce a new parameter; $\chi \in [0, 1]$, the crossing probability; to step 3 of the PG algorithm. We no longer reject crossings outright; instead, each time a crossing is encountered it is allowed with probability χ. For example, suppose a new edge crosses three existing ones, then the check will be applied three times and if each one is passed then the new edge is permitted.

Networks with $n = 10,000$, $m = 2$ and varying values of χ between 0.0 and 1.0 were grown and investigated using similar tools to those in the previous section. The relevant degree distributions are shown in figure 4a. For low values of χ the plots appear to be a reasonable fit for the power law observed in the case when $\chi = 0.0$. As we increase χ the plots become more akin to the exponential curve that we expect for the $\chi = 1.0$ case.

Figure 4b shows the average maximum degree observed in these experiments. Those plots with low values of χ again seem to be a closer to the calculated value for $\chi = 0.0$. Conversely, high values approach our estimate for the exponential case.

In figures 4c and 4d we examine how the assortativity and clustering vary with crossing probability. In the first case we see that a rises from -0.05 for $\chi = 0.0$ to 0.15 when $\chi = 1.0$. The curve of the plot is smooth although its rate of change is markedly more pronounced beyond $\chi = 0.4$. In the latter case the clustering, c falls from 0.49 when $\chi = 0.0$ to a negligible value when $\chi = 1.0$. The curve here is roughly linear with a lower gradient at the ends.

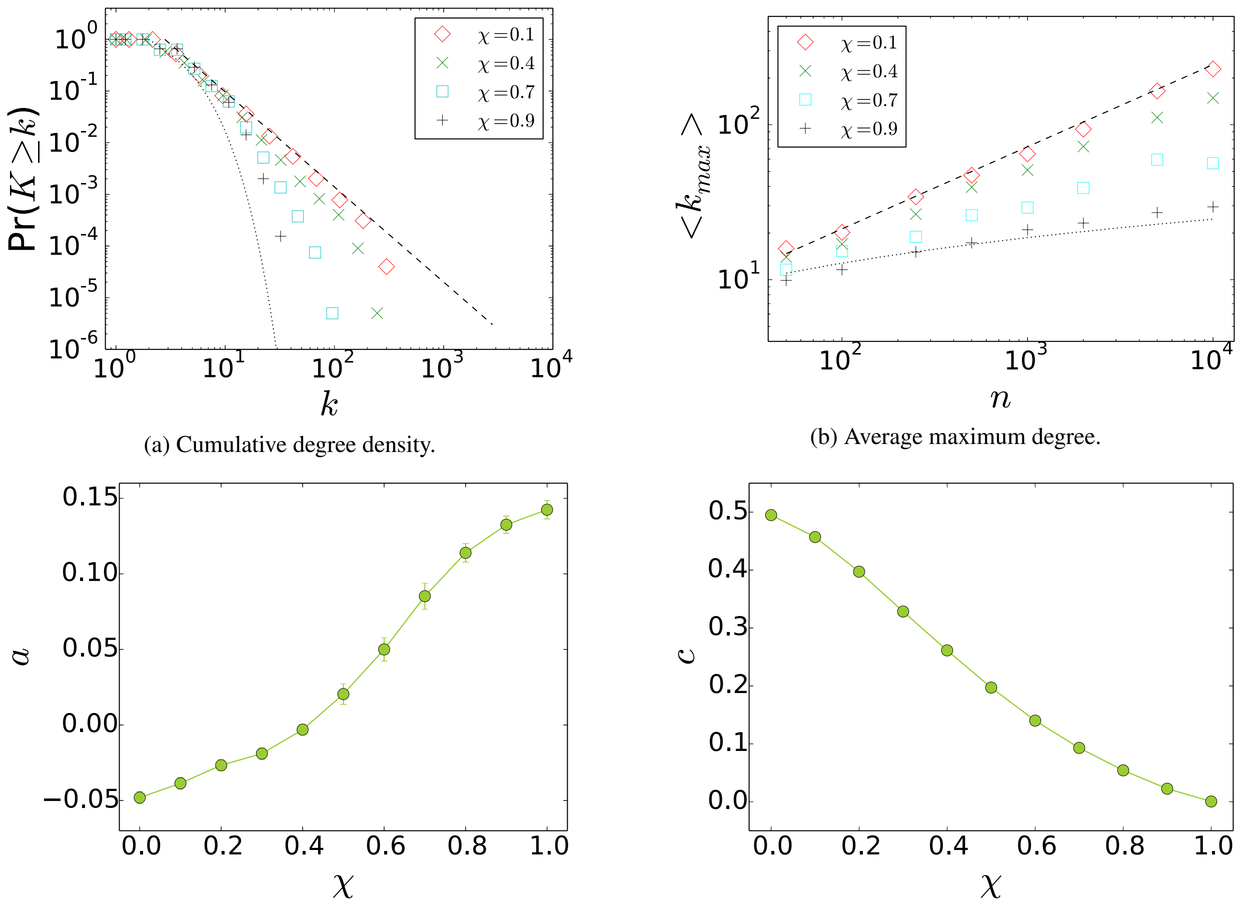
In summary we note that both the clustering and assortativity change smoothly across the χ parameter. We also highlight the degree density and average maximum degree plots where the individual curves for each value of χ appear in an ordered fashion between the fits for the $\chi = 0.0, 1.0$ cases. In light of this evidence we conclude that allowing edge crossing modifies the network from one with the PG properties to a uniform attachment model in a smooth, graded fashion with no abrupt transitions.

The values observed for assortativity and clustering in the $\chi = 1.0$ case are consistent with those obtained from a numerical simulation of a uniform attachment model. The negligible clustering is expected since the new edges are not subject to any pressure to attach to a node that would cause c to increase. As χ is decreased the effect outlined in the previous section takes hold and the clustering increases.

High assortativity in the uniform attachment model is due to an ageing effect, i.e. older edges will be both more likely to be connected to each other and to receive a higher amount of connections over the course of the simulation. Here, the effect of decreasing χ is to introduce more high degree hubs into the simulation thus driving the outcome towards the disassortativity outlined in the previous section.

Conclusion

In this paper we have outlined a minimal model of spatial network construction. It's principal features are stepwise



(a) Cumulative degree density.

(b) Average maximum degree.

(c) Assortativity.

(d) Clustering.

Figure 4: Cumulative degree distribution for networks created using planar growth with the crossing probability modification. Dashed line is the power law with exponent $\alpha_{m=2}$, the best fit of the $\chi = 0.0$ experiment (4a). Average maximum degree observed in the same experiments. Dashed and dotted lines are the same references plotted in figure 2b and are fits for the $\chi = 1.0$ and $\chi = 0.0$ cases (4b). Average assortativity observed in PG networks with the crossing probability modification and varying χ (4c). Average clustering (4d).

growth and the conservation of planarity at each stage of that growth. We have produced a small world network with a power law degree distribution that exhibited disassortativity and high clustering. We have also demonstrated that individually neither the mechanism of growth nor that of planarity alone can fully account for these properties. In each of the cases where one of these ingredients was removed while retaining the other, a network with an exponential degree distribution was observed.

We further established a connection with an existing model of network growth, the Random Apollonian network. This was achieved by modifying the original RAN algorithm so that it became equivalent to the Planar Growth process. The result also lent further weight to the claim that the resulting network was scale free.

Empirical examples of spatial networks tend to exhibit a degree distribution that is peaked rather than scale free. Further, these networks tend to have low assortativity with a high clustering coefficient (Barthélemy, 2011). Moreover, there is a need for models that act as a reference case for these cases since, as yet, there is no general agreement within the literature on one to use.

In this light we have investigated the consequences of relaxing planarity by allowing edge crossing with varying probability. We found that key network measures transitioned from those associated with absolute enforcement of planarity to those associated with no planarity. We have described a continuum of networks, tunable by the crossing probability parameter which allow us to explore considerable variation in their structure. In this way the model can be tuned to act as a plausible baseline for comparison between theoretical models.

References

Amaral, L. A. N., Scala, A., Barthelemy, M., and Stanley, H. E. (2000). Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21):11149–11152.

Andrade Jr, J. S., Herrmann, H. J., Andrade, R. F., and da Silva, L. R. (2005). Apollonian networks: Simultaneously scale-free, small world, euclidean, space filling, and with matching graphs. *Physical Review Letters*, 94(1):018702.

Balberg, I. (1985). Universal percolation-threshold limits in the continuum. *Physical review B*, 31(6):4053.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Barabási, A.-L., Albert, R., and Jeong, H. (1999). Mean-field theory for scale-free random networks. *Physica A: Statistical Mechanics and its Applications*, 272(1):173–187.

Barnett, L., Di Paolo, E., and Bullock, S. (2007). Spatially embedded random networks. *Physical Review E*, 76(5):056115.

Barthélemy, M. (2011). Spatial networks. *Physics Reports*, 499(1):1–101.

Barthélemy, M. and Flammini, A. (2008). Modeling urban street patterns. *Physical review letters*, 100(13):138702.

Bassett, D. S., Greenfield, D. L., Meyer-Lindenberg, A., Weinberger, D. R., Moore, S. W., and Bullmore, E. T. (2010). Efficient physical embedding of topologically complex information processing networks in brains and computer circuits. *PLoS Computational Biology*, 6(4):e1000748.

Ben-Avraham, D., F Rozenfeld, A., Cohen, R., and Havlin, S. (2003). Geographical embedding of scale-free networks. *Physica A: Statistical Mechanics and its Applications*, 330(1):107–116.

Brede, M. (2011). Growth and optimality in network evolution. *Artificial life*, 17(4):281–291.

Buesser, P. and Tomassini, M. (2012). Evolution of cooperation on spatially embedded networks. *Physical Review E*, 86(6):066107.

Bullock, S., Barnett, L., and Di Paolo, E. A. (2010). Spatial embedding and the structure of complex networks. *Complexity*, 16(2):20–28.

Cardillo, A., Scellato, S., Latora, V., and Porta, S. (2006). Structural properties of planar graphs of urban street patterns. *Physical Review E*, 73(6):066107.

Chan, S. H. Y., Donner, R. V., and Lämmer, S. (2011). Urban road networks, spatial networks with universal geometric features? *The European Physical Journal B-Condensed Matter and Complex Systems*, 84(4):563–577.

Clauset, A., Shalizi, C. R., and Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, 51(4):661–703.

Corson, F. (2010). Fluctuations and redundancy in optimal transport networks. *Physical Review Letters*, 104(4):048703.

Courtat, T., Gloaguen, C., and Douady, S. (2011). Mathematics and morphogenesis of cities: A geometrical approach. *Physical Review E*, 83(3):036106.

Dall, J. and Christensen, M. (2002). Random geometric graphs. *Physical Review E*, 66(1):016121.

Dorogovtsev, S., Mendes, J., and Samukhin, A. (2001). Size-dependent degree distribution of a scale-free growing network. *Physical Review E*, 63(6):062101.

Doye, J. P. and Massen, C. P. (2004). Self-similar disk packings as model spatial scale-free networks. *arXiv preprint cond-mat/0407779*.

Faloutsos, M., Faloutsos, P., and Faloutsos, C. (1999). On power-law relationships of the internet topology. In *ACM SIGCOMM Computer Communication Review*, volume 29, pages 251–262. ACM.

Gastner, M. T. and Newman, M. E. (2006). Shape and efficiency in spatial distribution networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(01):P01015.

Gudmundsson, A. and Mohajeri, N. (2013). Entropy and order in urban street networks. *Scientific reports*, 3.

Haruna, T. (2011). Global structure of directed networks emerging from a category theoretical formulation of the idea objects as processes, interactions as interfaces. In *Advances in Artificial Life, ECAL 2011, Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*, pages 310–317.

Hayashi, Y. (2006). A review of recent studies of geographical scale-free networks. *Information and Media Technologies*, 1(2):1136–1145.

Herrmann, C., Barthélemy, M., and Provero, P. (2003). Connectivity distribution of spatial networks. *Physical Review E*, 68(2):026128.

Huang, W., Chen, S., and Wang, W. (2014). Navigation in spatial networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 393:132–154.

Huson, M. L. and Sen, A. (1995). Broadcast scheduling algorithms for radio networks. In *Military Communications Conference, 1995. MILCOM'95, Conference Record, IEEE*, volume 2, pages 647–651. IEEE.

i Cancho, R. F., Janssen, C., and Solé, R. V. (2001). Topology of technology graphs: Small world patterns in electronic circuits. *Physical Review E*, 64(4):046119.

Jeong, H., Mason, S. P., Barabási, A.-L., and Oltvai, Z. N. (2001). Lethality and centrality in protein networks. *Nature*, 411(6833):41–42.

Jiang, B. (2007). A topological pattern of urban street networks: universality and peculiarity. *Physica A: Statistical Mechanics and its Applications*, 384(2):647–655.

Katifori, E., Szöllősi, G. J., and Magnasco, M. O. (2010). Damage and fluctuations induce loops in optimal transport networks. *Physical Review Letters*, 104(4):048704.

Kleinberg, J. M. (2000). Navigation in a small world. *Nature*, 406(6798):845–845.

Lee, S. H. and Holme, P. (2012). Exploring maps with greedy navigators. *Phys. Rev. Lett.*, 108:128701.

Lever, J. J., Nes, E. H., Scheffer, M., and Bascompte, J. (2014). The sudden collapse of pollinator communities. *Ecology letters*, 17(3):350–359.

Levinson, D. (2012). Network structure and city size. *PloS One*, 7(1):e29721.

Lotker, Z. and Peleg, D. (2010). Structure and algorithms in the sinr wireless model. *ACM SIGACT News*, 41(2):74–84.

Louf, R., Jensen, P., and Barthelemy, M. (2013). Emergence of hierarchy in cost-driven growth of spatial networks. *Proceedings of the National Academy of Sciences*, 110(22):8824–8829.

Manna, S. S. and Sen, P. (2002). Modulated scale-free network in euclidean space. *Physical Review E*, 66(6):066114.

Maslov, S., Sneppen, K., and Zaliznyak, A. (2004). Detection of topological patterns in complex networks: correlation profile of the internet. *Physica A: Statistical Mechanics and its Applications*, 333:529–540.

Masucci, A., Smith, D., Crooks, A., and Batty, M. (2009). Random planar graphs and the London street network. *The European Physical Journal B-Condensed Matter and Complex Systems*, 71(2):259–271.

Masuda, N., Miwa, H., and Konno, N. (2005). Geographical threshold graphs with small-world and scale-free properties. *Physical Review E*, 71(3):036108.

Milgram, S. (1967). The small world problem. *Psychology Today*, 2(1):60–67.

Miralles, A., Comellas, F., Chen, L., and Zhang, Z. (2010). Planar unclustered scale-free graphs as models for technological and biological networks. *Physica A: Statistical Mechanics and its Applications*, 389(9):1955–1964.

Mukherjee, G. and Manna, S. (2006). Weighted scale-free networks in euclidean space using local selection rule. *Physical Review E*, 74(3):036111.

Newman, M. E. (2001). The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409.

Newman, M. E. (2003). The structure and function of complex networks. *SIAM review*, 45(2):167–256.

Park, J. and Newman, M. E. (2003). Origin of degree correlations in the internet and other networks. *Physical Review E*, 68(2):026112.

Quintanilla, J., Torquato, S., and Ziff, R. M. (2000). Efficient measurement of the percolation threshold for fully penetrable discs. *Journal of Physics A: Mathematical and General*, 33(42):L399.

Rozenfeld, A. F., Cohen, R., Ben-Avraham, D., and Havlin, S. (2002). Scale-free networks on lattices. *Physical Review Letters*, 89(21):218701.

Rui, Y., Ban, Y., Wang, J., and Haas, J. (2013). Exploring the patterns and evolution of self-organized urban street networks through modeling. *The European Physical Journal B*, 86(3):1–8.

Salathé, M., Kazandjieva, M., Lee, J. W., Levis, P., Feldman, M. W., and Jones, J. H. (2010). A high-resolution human contact network for infectious disease transmission. *Proceedings of the National Academy of Sciences*, 107(51):22020–22025.

Strano, E., Nicosia, V., Latora, V., Porta, S., and Barthélemy, M. (2012). Elementary processes governing the evolution of road networks. *Scientific Reports*, 2.

Tan, H., Peng, M., Tse, C. K., and Wu, F. (2014). Global similarity tests of physical designs of circuits: A complex network approach. *Applied Mathematics and Computation*, 230:96–103.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of small-worldnetworks. *Nature*, 393(6684):440–442.

Xie, F. and Levinson, D. (2007). Measuring the structure of road networks. *Geographical Analysis*, 39(3):336–356.

Zhou, T., Yan, G., and Wang, B.-H. (2005). Maximal planar networks with large clustering coefficient and power-law degree distribution. *Physical Review E*, 71(4):046141.

Emotional modulation of peripersonal space impacts the way robots interact

Marwen Belkaid, Nicolas Cuperlier and Philippe Gaussier

Neurocybernetic team, ETIS laboratory, UMR 8051, CNRS/ENSEA/UCP
95000 Cergy Cedex, France.
{*firstname.lastname*}@ensea.fr

Abstract

Peripersonal space refers to the area around the body that is perceived as secure and reachable. The ability to build such a representation is necessary in both approach and avoidance behaviors. Several studies show that the perception of reachable and comfort areas depends on emotions. In this paper, we describe how we model an appetitive and an aversive pathway based on the role of some brain regions. The obtained emotional states modulate the robot perception of its peripersonal space. This representation is directly used to control the robot behavior. Based on a single-resource multirobot experiment, we show the impact of such an emotional modulation. Aggressive or fearful behaviors emerge from the dynamics of interaction between the simulated robots.

Introduction

Peripersonal space (PPS) refers to a multimodal sensorimotor interface between the body and its environment (Rizzolatti et al., 1997). It is the area around individuals in which an external intrusion can be perceived as possibly threatening, or at least uncomfortable (Kennedy et al., 2009) (Tajadura-Jiménez et al., 2011). In addition, it also determines the reachable space in action-related contexts (Valdés-Conroy et al., 2012). Thus, PPS perception is by definition relevant in both approach and avoidance behaviors. The parietal cortex is thought to play a key role in such a multimodal representation of the surrounding (Rizzolatti et al., 1997)(Graziano and Cooke, 2006).

It has been demonstrated that PPS representation is plastic. Indeed, positively valenced objects tend to be perceived as more reachable than negative ones (Valdés-Conroy et al., 2012). However, the presence of threatening objects in our peripersonal area can be perceived differently. For instance, a knife seems farther when oriented toward us, i.e. when potentially dangerous (Coello et al., 2012). On the other hand, a positive affective state, induced by pleasant music for instance, can impact the PPS as well, reducing the area needed to feel confortable in over-crowded spaces (Tajadura-Jiménez et al., 2011).

In this work, we model an appetitive and an aversive pathway based on the idea that, in biological organisms, basic motivated behavior can be represented in terms of approach and avoidance. Following a constructivist approach, we mimic the role of brain regions involved in the emotion circuitry. Our aim is to have the minimal set of signals required to model the robot emotional state.

Various approaches for modeling emotions in robots and artificial agents can be found in the literature. They generally rely on a set of drives to guide agents behavior (Cañamero, 1997)(Hirth et al., 2007). Recent work also proposed models based on hormonal modulation (Lones et al., 2014) and neuromodulatory signals (Krichmar, 2013). Moreover, in some cases, emotions are considered as a way to implement metacontrollers and self-regulation loops (Sanz et al., 2013)(Jauffret et al., 2013).

Here, we want to model PPS as a representation of the surrounding area that is both secure and reachable. This representation should be modulated by the robot emotional states in order to integrate a subjective and motivated perception of its environment. More specifically, we address the cases where approach and avoidance motivations are in contradiction. Thereby, we can observe the impact of the emotional state on the robot behavior. We show the effect of PPS emotional modulation on the interaction between two robots in a single-resource situation. We also observe that their behavior expresses aspects of their internal states. Depending on whether lateral inhibition between appetitive and aversive pathways is allowed, the robots seem aggressive/determined or fearful/patient.

In the next sections, we present our model for the emotional modulation of peripersonal space and discuss results from multirobot competition simulations.

Modeling appetitive and aversive pathways

Pleasure and pain are considered as basic components of emotions (Damasio, 2003). The dopaminergic pathways are generally associated to pleasure and reward prediction in the literature (Berridge, 2012). The Ventral Tegmental Area (VTA) contains the largest group of dopamine neurons and projects to limbic structures like the amygdala (mesolimbic pathway) and prefrontal cortex (mesocortical pathway). On the other hand, nociceptors signals are transmitted from the spinal cord. In the neural processing of pain and punishment aversion, serotonine, which is mainly produced in the Raphe Nuclei, also plays a significant role (Cools et al., 2008). In this paper, we do not aim at a detailed model of the pleasure and pain neural circuitries. However, we are interested in integrating reward and punishment signals and modeling interactions between dopaminergic and serotonergic pathways to inhibit either appetitive or aversive behaviors.

Motivations are also a key component of emotions (Damasio, 2003). Some theorists see the latter as an expression of motivational states that prepares for action and triggers cognitive control (Pessoa, 2008) (Michael Inzlicht and Bruce D. Bartholow and Jacob B. Hirsh, 2015). Here, we are interested in modeling low-level appetitive and aversive drives. For example, the Hypothalamus (HTH) links the nervous system to the endocrine system. Thereby, it intervenes in various bodily functions such as monitoring physiological parameters and regulating hunger and thurst. Moreover, one of the function of the superior colliculus (SC) is the integration of multiple sensory input in order to trigger defensive behavior like avoidance or withdrawal (Comoli et al., 2012).

Inputs from hypothalamus and sensory information are relayed from thalamus to amygdala. The latter plays a key role in emotions. It responds with higher activations in the presence of arousing stimuli and projects to the Reticular Formation (RF), which is thought to modulate the arousal level of the central nervous systems (Cardinal et al., 2002). In addition, (Kennedy et al., 2009) suggests the amygdala is necessary for PPS representation. Indeed, it is required for the attribution of positive or negative values to stimuli through stimulus-stimulus (S-S) Pavlovian conditioning. It also allows for stimulus-response (S-R) Pavlovian conditioning (Cardinal et al., 2002). In this work, we only describe reflex pathways. However, our model is consistent with the idea that the amygdala participates in building the robot emotional and motivational states through conditioning. For instance, unconditional stimuli can be associated with reward or punishment signals. Such predictions would trigger or emphasize approach or avoidance behavior like in the incentive motivation literature (Berridge, 2012). Figure 1 summarizes the way we model appetitive and aversive pathways in order the modulate robot PPS perception. Please note that we do not aim to present a precise model of the brain structures involved. We rather mimic some of their functions in order to propose a bio-inspired model that is consistent with the literature.

In this work, we model two basic low-level motivations: the feeding drive (appetitive) and the safety drive (aversive). In the latter, we calculate the mean activity on the n_s proximity sensors s_i to obtain the level th of threat at time t:

$$th(t) = \frac{\sum_i s_i}{n_s} \tag{1}$$

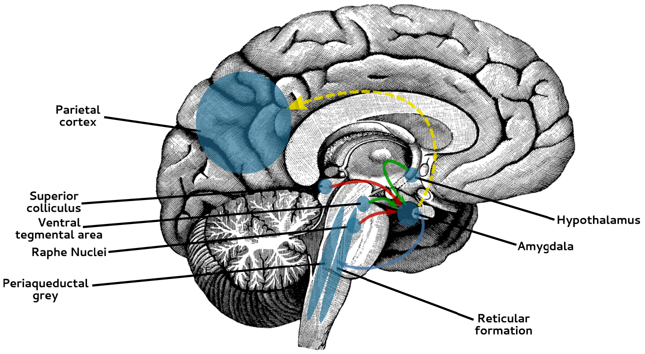


Figure 1: Brain regions involved in our model. The appetitive and aversive pathways are respectively represented in green and red. The yellow arrow illustrates the emotional modulation of the peripersonal space perception.

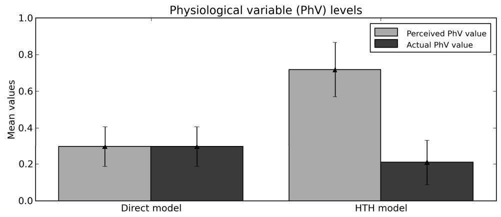


Figure 2: Comparison between a direct perception of the level of a physiological variable (PhV) and the HTH model (30 feeding cycles each). We suppose the PhV increases and decreases linearly in our system. The results shows how using the HTH model allows for anticipating the lack of food and prevents from depletion.

The feeding drive is guided by the preceived level of a simulated physiological variable, based on the model of hypothalamus proposed in (Hasson, 2011). The level r of the physiological variable associated to the resource at time t is:

$$r(t) = \alpha_r(r_{max} - r(t-dt)).I(t) - \beta_r r(t-dt)(1-I(t)) \tag{2}$$

where r_{max} is the maximal variable level (set to 1), α_r and β_r respectively indicate the ingestion and the consumption speed factors and I is the ingestion signal. Using this HTH model gives the robot the ability to anticipate the lack of food in order to trigger the appropriate behavior. For the sake of simplicity, let us consider the level of a physiological variable (PhV) increases and decreases linearly. Considering both functions reach the maxima simultaneously, with the HTH model, the perceived PhV level drops below the satisfaction level more quickly in the consumption phases. Figure 2 shows the result of such a comparison between a direct perception of the PhV level and the HTH model.

We define the approach m_{ap} and avoidance m_{av} motiva-

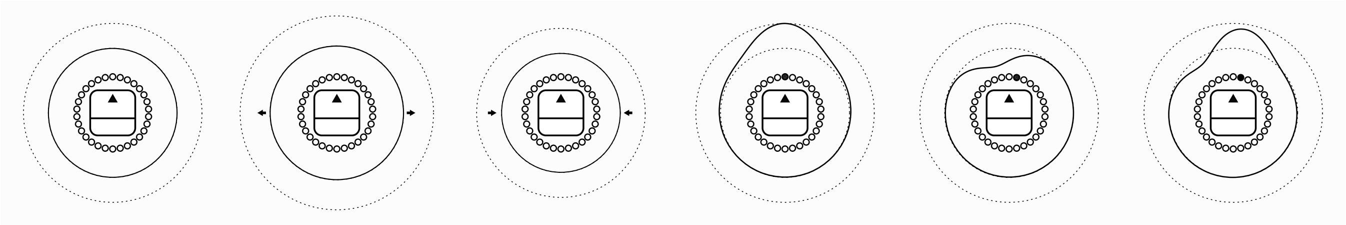


Figure 3: Different forms of modulation of robot PPS. FAR-LEFT: No modulation. LEFT and CENTER-LEFT: The comfort zone contracts or dilates according to the pleasantness of the affective state. CENTER-RIGHT, RIGHT, and FAR-RIGHT: Also, appetitive and aversive stimuli respectively induce an extension or a retraction of the reachable space in the corresponding direction.

tion levels at time t as following:

$$\begin{aligned} m_{ap}(t) &= \varepsilon_m.m_{ap}(t-dt) - \gamma_m.m_{av}(t-dt) \qquad (3) \\ m_{av}(t) &= \varepsilon_m.m_{av}(t-dt) - \gamma_m.m_{ap}(t-dt) \end{aligned}$$

where ε_m and γ_m respectively represent the integration and inhibition factors of the competition.

Similarly, we obtain a medium-term affective state a that integrates punishment a_{pn} and reward a_{rw} signals at time t using the following equations :

$$\begin{aligned} a(t) &= a_{rw}(t) - a_{pn}(t) \qquad with \qquad (4) \\ a_{pn}(t) &= \varepsilon_a.a_{pn}(t-dt) - \gamma_a.a_{rw}(t-dt) \\ a_{rw}(t) &= \varepsilon_a.a_{rw}(t-dt) - \gamma_a.a_{pn}(t-dt) \end{aligned}$$

where ε_a and γ_a respectively represent the integration and inhibition factors of the competition.

Proposed model for emotional modulation of peripersonal space

As a sensorimotor interface with the world related to both approach and avoidance behaviors, we suggest it is interesting to model PPS in a robotic system. Here, we are more precisely interested in its modulation by emotional states. If we consider a mobile robot in a navigation task, we can represent various states of its PPS perception like in Fig. 3. Indeed, its comfort zone can contract or dilate according to the pleasantness of the current affective state. Also, appetitive and aversive stimuli respectively induce an extension or a retraction of the reachable space in the corresponding direction.

We propose that peripersonal space perception is based on a working memory that integrates sensorimotor information (See Fig. 4). For instance, the robot can remember the position of an obstacle it avoided. Also, it can update a path integration vector associated with a goal according to the speed and direction of instantaneous movement. We propose this sensorimotor input has to be integrated according to the current affective state. Thus, if the robot perceives a collision as a punishment signal, obstacles become more salient and leave a bigger trace in the working memory. Indeed, punishment-induced negative state expands the robot comfort zone, i.e. the space in which intrusions seems threatening.

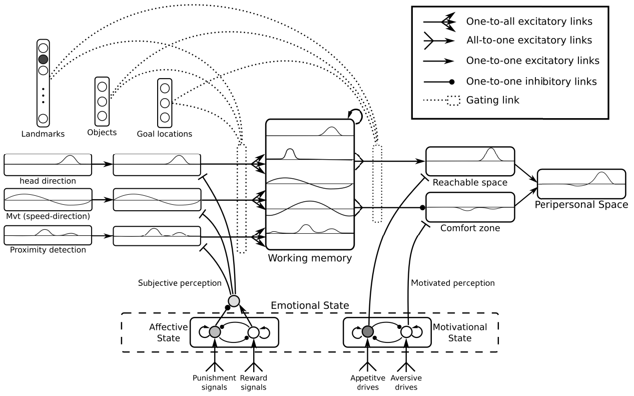


Figure 4: Model for building a representation of the robot peripersonal space. It is based on working memory taking input from various sensory modalities. PPS is modulated by the robot emotional states in order to integrate a subjective and motivated perception of its environment.

In this paper, we do not focus on the building of the working memory. It is based on the principle described in (Hasson and Gaussier, 2010). The robot can learn to associate several goal locations to the drives they satisfy (e.g. hunger and thurst). Proprioceptive path integration fields allows it to return to the resource locations when needed. However, please note that the working memory has limited capacity. Yet the model is able to handle multiple goals by replacing the least used memory field when new resources are discovered.

Information from the working memory can be merged to offer a representation of the robot peripersonal space. However, this perception highly depends on the motivational state. For example, an appetitve drive make a desirable object seem more reachable. Likewise, a defensive motivation highlights aversive stimuli in the comfort zone and generates an avoidance behavior. Therefore, we suggest a second emotional modulation occurs in order to filter information coming from the working memory. This motivated perception is directly used to determine robot actions.

Single-resource multirobot competition

Implementation details

This experiment is performed on the Webots simulator in order to avoid damaging real robots. We simulate two identical robots moving in a 17.5 m x 15 m environment. The 4-wheel mobile robot platforms are 40 cm-wide, 50 cm-long and 1 m-high. They are embedded with light sensors. The latter are placed under the robots to detect a 45 cm x 45 cm colored zone in the center which is considered as a resource. The platform also has 9 ultra-sound proximity sensors, of which we only use a subset to cover a 180 degrees-wide front field.

The robots affective states depend on the received punishment and rewards signals used to simulate pain and pleasure (See Equation 4). In this experiment, they are respectively given by collision and resource detection. In addition, robots have one appetitive and one aversive drive – respectively feeding and protecting its own physical integrity (See Equations 2 and 1).

Starting from the resource location, the return vector is calculated by integrating the speed and direction of instantaneous movements. The activity p of each neuron i in the path integration field at time t is given by the following equation:

$$p_i(t) = \sum_{t_r}^{t} (s(t).cos(\frac{d(t) + 2\pi i/n}{n})).(1 - R(t)) \qquad (5)$$

where t_r is the last reset time, s the linear speed, d the direction, R the reset signal and n the size of the neural field.

The feeding drive becomes active whenever the level of the physiological variables drops below a satisfaction threshold s_{th}. The robot then uses the path integration vector to return to the resource. Similarly, obstacle detection triggers the defensive drive and generates an avoidance behavior. In some case, these two low-level motivations can be contradictory, e.g. if there is an obstacle (object or other robot) on the way. A competition between the appetitive and aversive drives allows them to inhibit each other, which favours the approach or the avoidance behavior depending on the drives levels (See Equations 4).

We use a dynamic neural field (DNF) (Schöner et al., 1995) to merge reachable space and comfort zone information. Appetitive and aversive stimuli given as input generate attractors and repulsors in the DNF. The potential u of each neuron x is updated as following:

$$\tau.\frac{u(x,t)}{dt} = -u(x,t) + I(x,t) + h + \int_{z \in V_z} w(z).f(u(x-z,t)).dz \qquad (6)$$

where $f(x) = tanh(x)$ and is used to calculate the neuron activity, τ is the time constant, I the input, h a constant inhibition potential and w an interaction kernel. Using a DoG (Difference of Gaussian) function as the interaction kernel allows proximal stimuli to reinforce each other and to inhibit distant ones. The highest neuron activity is used to calculate the linear speed. Also, a readout of the output derived signal (according to the current orientation) allows for computing the rotational speed.

In this experiment, $s_{th} = 0.5, \varepsilon_m = 0.8, \gamma_m = 0.2, \varepsilon_a = 0.2, \gamma_m = 0.8, \alpha_r = 0.1$ and $\beta_r = 0.01$.

Method

In order to study the impact of the emotional modulation of the peripersonal space, we compare our model behavior with two altered versions of the architecture:

- *version* We use the proposed model described in the previous section.
- *NoCompet version* In this version, we still modulate robot PPS according to its emotional state. However, there is no lateral inhibition between punishment and reward signals nor between appetitive and aversive drives.
- *NoModul version*: In this version, no modulation of approach/avoidance is performed at all. Robot drives only serve for triggering homing behavior for example. This version is the closest to a classical reactive architecture. Except here approach and avoidance have the same weight in the DNF.

We define a cycle as an interval in which a robot, initially satisfied (non-hungry), consumes the energy obtained from the previous ingestion and returns to the resource in order to feed once again. Each of these cycles is considered as an independant sample of the multirobot competition for the resource. Once the feeding drive satisfied, robots get away from the resource. They randomly navigate in the environment updating their path integration field to be able to return to the resource when needed. We use three measures:

- *min_phyvar*: Lowest level of the physiological variable associated with the feeding drive,
- *nb_own_access*: Number of own accesses to the resource within a full cycle,
- *nb_other_access*: Number of other robot accesses to the resource within own cycle.

The first one is a measure of food depletion, i.e. how close to starvation the robots get. The two latter quantify cycle interruptions. Besides, we consider two variables:

- *model*: Whether our model is used or not (the NoCompet and NoModul version are gathered in the same group),
- *version*: Which version is used (each of the 3 versions is association with a group).

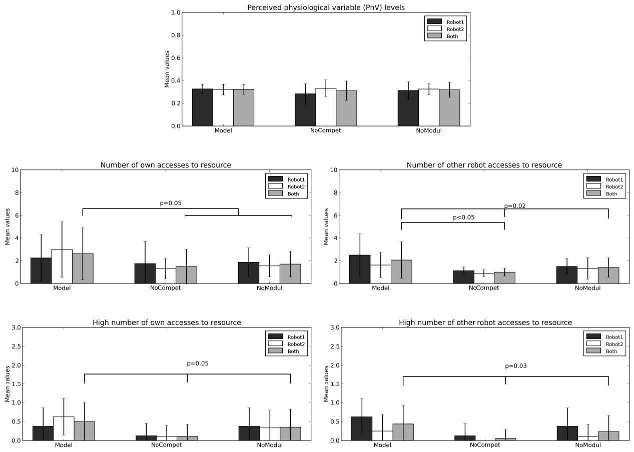


Figure 5: Statistical significance of the effect of different architecture versions on the considered measures. No effect has been revealed on food depletion (*min_phyvar*). But, a strong tendency is found with *model* variable on *nb_own_access* and with *version* on *bin(nb_own_access)*. There is also a main effect of *version* on *nb_other_access* and *bin(nb_other_access)*.

Results

Statistical study First, we compare the three architecture versions in 15-minute simulations (N=51, N_{Model}=16, $N_{NoCompet}$=18, $N_{NoModul}$=17; *min_phyvar*: μ=0.68, σ=0.07, ; *nb_own_access*: μ=1.92, σ=1.78 *nb_other_access*: μ=1.47, σ=1.14). None of the variable follows a normal distribution.

Kruskal-Wallis non-parametric test shows that there is no effect of *version* nor *model* on *min_phyvar* (resp. $Chi^2 = 0.45, p = 0.80$; and $Chi^2 = 0.00, p = 0.95$). No significant effect of *version* on *nb_own_access* was found either ($Chi^2 = 5.36, p = 0.07$). However, there is a strong tendency with *model* ($Chi^2 = 3.81, p = 0.05$). Also, there is a main effect of both *model* and *version* on *nb_other_access* (resp. $Chi^2 = 6.03, p = 0.01$; and $Chi^2 = 8.19, p = 0.02$). In addition, Mann-whitney test shows a significant effect of *version* on *nb_other_access* between *Model* and *NoCompet* ($U = 82, p < 0.05$).

Moreover, let us consider two additional measures *bin(nb_own_access)* and *bin(nb_other_access)* respectively equal to 1 if *nb_own_access* and *nb_other_access* are greater than 1, and 0 otherwise. Indeed, in the case of a perfect alternation of the robots over the resource, each should access it exclusively and only once in every cycle. Any different configuration could correspond to a feeding cycle being interrupted by another robot. In this case, we find a strong tendency on *bin(nb_own_access)* with both *model* and *version* (resp. $Chi^2 = 3.68, p = 0.05$; and $Chi^2 = 6.01, p = 0.05$) as well as a significant effect on *bin(nb_other_access)* (resp. $Chi^2 = 5.19, p = 0.02$; and $Chi^2 = 6.73, p = 0.03$).

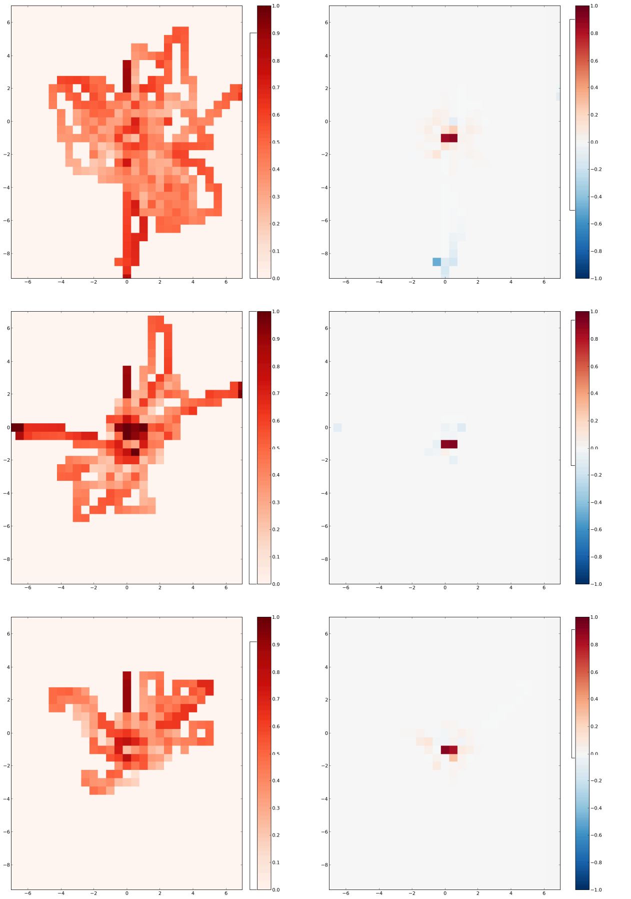


Figure 6: Heatmaps representing arousing (left column) and positively and negatively valenced (right column) areas in the environment. They are averaged for both robots. The brackets on the colorbars show the intervals between min and max values. The lines respectively correspond to *Model*, *NoCompet* and *NoModul* from top to bottom.

Behavioral comparison When the *NoCompet* or *NoModul* architectures are used, the robots tend to be unable to access the resource before it is free – i.e. before the other robot is done feeding. However, with the *Model* version, they can push one another and compete for the resource. This is due to the approach sub-behavior inhibiting the defensive one.

Figure 6 shows arousing areas as well as positively and negatively valenced ones in the environment. We notice that in the *NoCompet* case, the area around the resource is one of the most arousing because both drives are simultaneously active. The robots generally need to feed but avoid collisions with the other one currently on the resource. Also, rewards are only obtained right on the center of the resource and punishments around it. However, with the *Model* and *NoModul* architectures, the emotional states are generated exactly in the same way even though, in the latter, they do not modulate robots PPS; and thereby their behavior. We see that most arousing areas are less localized than in the *NoCompet* case. Indeed, the competition between approach and aversion makes them inhibit each other and avoids arousal saturation. But, with the *Model* version, negative valence reaches a lower level than with the two other architectures. This is due to the robot tendency to avoid collisions less than with *NoCompet* and *NoModul*.

General discussion

In this experiment, we showed how the emotional modulation of robots PPS impacted their behavior and the way they interacted. We compared our *Model* with two altered versions of the architecture. Statistical results regarding the *min_phyvar* measure revealed no significant difference between the architectures in terms of food depletion. This is due to the random exploration following feeding phases and to resource consumption being slower that its ingestion. This leaves the possibility for the robot to alternate in resource access. Yet, it is interesting to observe how this alternation occurs. That is to say, how the robots interact in this survival task.

The measures *nb_own_access* and *nb_other_access* allow for caracterizing this alternation in resource access. Indeed, if robots wait for each other, each one has access to the resource exclusively and only once in every cycle. If greater than 1, they indicate that a feeding phase was interrupted. The results show a strong tendency with the variable *model* on *nb_own_access* and with both *model* and *version* on *bin(nb_own_access)*. They also show a significant effect of both variables on *nb_other_access* and *bin(nb_other_access)*.

Cycle interruption are directly related to robots being pushed by each other far from the resource. When the *NoCompet* or *NoModul* architectures are used, robots tend to be unable to access the resource before it is free. In the *NoModul* case, accidental collisions may occur but robots generally deviate from the resource in order to avoid the other one which is currently feeding. On the other hand, interactions between the robots seem to carry a social significance with *NoCompet* and *Model* versions. In both cases, they behave in a way that expresses aspects of their internal states. With the former, robots seem either patient or fearful. Their modulation of their PPS make them extend their comfort zone. They are more sensitive to aversive stimuli and defensive sub-behavior tend to take over the appetitive one. On the contrary, using the *Model* version, the robots seems more proactive and determined. When the resource is not available they try to push whatever is on their way. Thereby, they display an aggressive behavior.

Similar behaviors are observed in (Lones et al., 2014). A hormone-based model is tested in competitive and non-competitive environments. In the former case, aggressive or withdrawn populations of agents can emerge from an epigenetic adaptation mechanism. In the literature, a dichotomy of aggression opposes the proactive forms to the reactive ones (Weinshenker and Siegel, 2002)(Vitiello and Stoff, 1997). In the first class, a predatory attack serves as a way to get a reward. It is instrumental and is generally accompanied by a very low level of sympathetic arousal. This kind of behavior is not modeled here, although it can be observed due to the dynamics of interaction between the robots. In contrast, affective defense describes any aggressive response triggered by elements of fear and/or threat. It is reactive, rather than proactive or instrumental. Also, it is more related to anger, which is represented as a negatively valenced affect with relatively high arousal (or intensity) in dimensional models of emotion (Posner et al., 2005). The emotional state of our robot during aggressive episodes is consistent with this representation.

Unlike (Lones et al., 2014), in our work, the emergence of aggressive vs. fearful behavior depends on a lateral inhibition between appetitive and aversive motivations being allowed or not in the architecture. Although we take inspiration from interactions between dopaminergic and serotonergic pathways, the competition between approach and avoidance implemented here is relatively basic. Indeed, in some context, such a winner-takes-all mechanisms would be inefficient or irrelevant. (Krichmar, 2013) highlights the role of cholinergic and noradrenergic systems in regulating the dopamine and serotonin levels. Modeling such topdown neuromodulatory mechanisms could be an interesting option.

Besides, (Kennedy et al., 2009) shows that individuals with bilateral amygdala lesions fail to represent peripersonal space boundaries correctly. The authors suggest this is due to the absence of strong emotional responses to personal space violation. Indeed, the capacity to associate stimuli to reward or punishment signal seems necessary. (Berridge, 2012) also highlights the role of Pavlovian conditioning in reward prediction and incentive motivation. In our case for instance, adding instrumental learning mechanisms would allow our robots to switch from purely reactive affective responses to goal-oriented aggression.

One particularity of our model is the idea that PPS representation is based on a subjective perception. The size of robot comfort zone is modulated by its affective state (Tajadura-Jiménez et al., 2011). This accentuates the fearful behavior for example. One could argue it is very specific to the defensive mechanisms. Yet, its perception of its elementary displacements could also be modulated likewise. In most cases where feeding only depends on the ability to return to a resource like here, this would lead path integration to failure. However, future work will investigate situations where such an erroneous perception can be useful.

Conclusion

In conclusion, we propose a model allowing the robot to build a representation of its peripersonal space based on a subjective and motivated perception. Based on the role of certain brain structures, we indentify a set of signals required to model the robot emotional state. Thus, in our model, signals of pleasure vs. pain and approach vs. avoidance interact and modulate the robot perception. By simulating a simplified version of biological behaviors, we are able to observe the impact of some alterations on the model. We show how the emotional modulation of the peripersonal space makes the robots interact in a way that expresses aspects of their internal states. In addition, depending on whether lateral inhibition between appetitive and aversive pathways is allowed, the robots seem more aggressive or more fearful.

References

Berridge, K. C. (2012). From prediction error to incentive salience: Mesolimbic computation of reward motivation. *European Journal of Neuroscience*, 35(7):1124–1143.

Cañamero, D. (1997). Modeling motivations and emotions as a basis for intelligent behavior. In *Proceedings of the first international conference on Autonomous agents AGENTS 97*, number Abbott 1884, pages 148–155. ACM Press.

Cardinal, R. N., Parkinson, J. A., Hall, J., and Everitt, B. J. (2002). Emotion and motivation: The role of the amygdala, ventral striatum, and prefrontal cortex.

Coello, Y., Bourgeois, J., and Iachini, T. (2012). Embodied perception of reachable space: how do we manage threatening objects?

Comoli, E., Das Neves Favaro, P., Vautrelle, N., Leriche, M., Overton, P. G., and Redgrave, P. (2012). Segregated Anatomical Input to Sub-Regions of the Rodent Superior Colliculus Associated with Approach and Defense.

Cools, R., Roberts, A. C., and Robbins, T. W. (2008). Serotoninergic regulation of emotional and behavioural control processes. *TRENDS in Cognitive Sciences*, 12(1).

Damasio, A. R. (2003). *Looking for Spinoza: Joy, sorrow, and the feeling brain.*

Graziano, M. S. A. and Cooke, D. F. (2006). Parieto-frontal interactions, personal space, and defensive behavior (DOI:10.1016/j.neuropsychologia.2005.09.009).

Hasson, C. (2011). *Modeling emotional mechanisms for an autonomous robot: A developmental and social perspective*. PhD thesis.

Hasson, C. and Gaussier, P. (2010). Path integration working memory for multi tasks dead reckoning and visual navigation. In *Proceedings of the 11th international conference on Simulation of adaptive behavior: from animals to animats*, pages 380–389. Springer-Verlag.

Hirth, J., Braun, T., and Berns, K. (2007). Emotion Based Control Architecture for Robotics Applications. In *Proceedings of the 30th annual German conference on Advances in Artificial Intelligence*, pages 464–467. Springer-Verlag.

Jauffret, A., Belkaid, M., Cuperlier, N., Gaussier, P., and Tarroux, P. (2013). Frustration as a way toward autonomy and self-improvement in robotic navigation. In *2013 IEEE 3rd Joint International Conference on Development and Learning and Epigenetic Robotics, ICDL 2013 - Electronic Conference Proceedings*.

Kennedy, D. P., Gläscher, J., Tyszka, J. M., and Adolphs, R. (2009). Personal space regulation by the human amygdala. *Nature neuroscience*, 12(10):1226–1227.

Krichmar, J. L. (2013). A neurorobotic platform to test the influence of neuromodulatory signaling on anxious and curious behavior. *Frontiers in neurorobotics*, 7(February):1.

Lones, J., Lewis, M., and Cañamero, L. (2014). Hormonal modulation of interaction between autonomous agents. In *2014 IEEE 4th Joint International Conference on Development and Learning and Epigenetic Robotics, ICDL-Epirob 2014*.

Michael Inzlicht and Bruce D. Bartholow and Jacob B. Hirsh (2015). Emotional foundations of cognitive control. *Trends in Cognitive Sciences*, 19(3):126–132.

Pessoa, L. (2008). On the relationship between emotion and cognition. *Nature Reviews Neuroscience*, 9(2):148–158.

Posner, J., Russell, J. A., and Peterson, B. S. (2005). The circumplex model of affect: an integrative approach to affective neuroscience, cognitive development, and psychopathology. *Development and psychopathology*, 17(3):715–734.

Rizzolatti, G., Fadiga, L., Fogassi, L., and Gallese, V. (1997). The space around us. *Science*, 277(5323):190–191.

Sanz, R., Sánchez-Escribano, M. G., and Herrera, C. (2013). A model of emotion as patterned metacontrol. *Biologically Inspired Cognitive Architectures*, 4:79–97.

Schöner, G., Dose, M., and Engels, C. (1995). Dynamics of behavior: Theory and applications for autonomous robot architectures.

Tajadura-Jiménez, A., Pantelidou, G., Rebacz, P., Västfjäll, D., and Tsakiris, M. (2011). I-space: The effects of emotional valence and source of music on interpersonal distance. *PLoS ONE*, 6(10).

Valdés-Conroy, B., Román, F. J., Hinojosa, J. a., and Shorkey, S. P. (2012). So far so good: emotion in the peripersonal/extrapersonal space. *PloS one*, 7(11):e49162.

Vitiello, B. and Stoff, D. M. (1997). Subtypes of aggression and their relevance to child psychiatry. *Journal of the American Academy of Child and Adolescent Psychiatry*, 36(3):307–315.

Weinshenker, N. J. and Siegel, A. (2002). Bimodal classification of aggression: Affective defense and predatory attack.

Simulating self-replicating, chemically immersed, microchip swarms

Uwe Tangen[1,3], Harold Fellermann[2,1] and Steen Rasmussen[1,4]

[1]Center for Fundamental Living Technology (FLinT), University of Southern Denmark, DK-5230 Odense, Denmark
[2]School of Computing Science, Newcastle University, NE1 7RU, Newcastle upon Tyne, UK,
[3]BioMIP, Organic Chemistry I, Ruhr-Universität Bochum, 44780 Bochum, Germany
[4]Santa Fe Institute, New Mexico, USA
tangen@sdu.dk, harold.fellermann@ncl.ac.uk, steen@sdu.dk

Silicon microchips of size 100x100 μm^2 with active electronic sensing and actuation reacting in a chemical environment is the vision of the research project http://www.micreagents.eu, see figure 1a. These chips, called lablets, are fabricated in standard $180nm$ silicon technology and will be equipped with custom developed supercapacitors allowing them to react autonomously in fluids. They are designed to catalyze reactions with inbuilt electrodes and utilize electro-osmotic flow to propel along in the fluidic system, McCaskill et al. (2012).

We explore swarms of solvated lablets in simulations using a developed simulation environment Chemical Swarm Language, CSL (2013), which essentially is a 2D reaction diffusion system with lablets as “active walkers”. Lablets can follow chemical gradients with a given propensity and have their own chemical reservoir. Chemical reactions are restricted to the onboard reservoir and are neglected outside of lablets. A state-machine is controlling the activation of lablet electrodes. Both chemicals and lablets diffuse with respective diffusion constants.

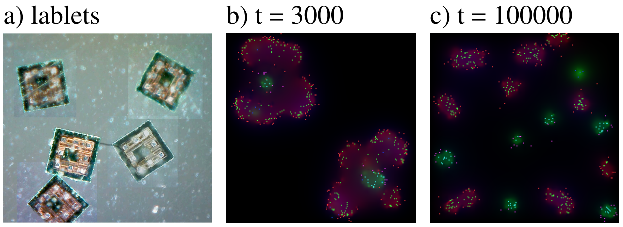


Figure 1: Picture a) shows a montage of microscope images of first generation CMOS $100\mu m$ lablets, which can be powered by an external field and show its activity via electro luminescence. In b) and c) the succesive division of two simulated lablet swarms is shown. Each dot in the image represents one lablet, the color depicts the actual state the lablets are in, and the colored clouds represent the chemicals in the spatial environment of size 200x200 lattice points. Lablets generating green and red chemicals repel from red and green chemicals respectively.

Emergent mesoscale structures are generated e.g. when situated lablets take up material in a low-concentration area and release it again in a high-concentration area. These structures can also develop if lablets catalyze reactions such that the same chemical(s) are depleted in low-concentration areas and produced in high-concentration locations. Result are dynamical steady-state structures, maintained by the active lablets powered by their onboard super capacitors. When the lablets can catalyze at least two chemicals, which lablets in different states try to avoid, the dynamical structures can divide into two or more new structures. Depending on the number of available lablets these colonies eventually grow and divide again, see time series in figure 1b below.

These highly nonlinear chemically induced structures are created as in quorum-sensing, Melke et al. (2010), via direct communication of the lablet swarm with the chemical environment. They also resemble self-replicating spots in the Gray-Scott model, Pearson (1993), although the mechanism being entirely different, and in contrast can be treated as robust autonomous mesoscale objects which e.g. can be dragged around. These objects represent chemically confined areas and as such offer a chemical reaction vessel, as in droplets or synthetic liposomes, Pereira de Souza et al. (2009), but do not suffer from the input-output issue with complex molecules or different phases. They represent a novel kind of protocells, Rasmussen et al. (2008) and promise entirely new applications in nonlinear chemistry.

Acknowledgements: This work is in part funded by the European Commission FP7 grant MICREAgents (318671).

References

McCaskill, J. et al. (2012). *IJUC*, 8:289–299.

Melke, P. et al. (2010). *PLoS Comput Biol*, 6:e1000819.

Pearson, J. E. (1993). *Science*, 261:189–192.

Pereira de Souza et al. (2009). *ChemBioChem*, 10(6):1056–1063.

Rasmussen, S. et al., editors (2008). *Protocells: Bridging Nonliving and Living Matter*. MIT Press, Boston.

CSL: https://github.com/harfel/MICREAgents.

Epigenetic inheritance speeds up evolution of artificial organisms

Yoram Vadée-le-Brun and Jonathan Rouzaud-Cornabas and Guillaume Beslon

Universite de Lyon, CNRS, INRIA Beagle team
INSA-Lyon, LIRIS, UMR5205, F-69621, France
guillaume.beslon@inria.fr

Abstract

DNA is not the sole medium by which parents transmit information to their offspring. Epigenetic inheritance, in particular, is based on the partial transmission of the cellular state of the parental cell to its descendants. Although the reality of epigenetic inheritance is now firmly established, whether it has an influence on the long term evolutionary process is still subject to debate.

To address this question, we used RAevol, an *in silico* experimental evolution platform, and defined 4 scenarios with static or dynamic environments and with or without epigenetic inheritance. Simulations in dynamic environments show that protein inheritance indeed increases the rate of evolution on the long term. But they also show that it impedes evolution in its very first stages. This negative effect can be explained by instabilities generated by the interference between the two inheritance mediums. On the opposite, the long term gain can be explained by protein inheritance reducing the constraints on the genetic regulation network.

Introduction

In biology, inheritance (or heredity) typically refers to the fact that offspring look more like their parents than like other random individuals. We call this kind of inheritance “phenotypic inheritance”. Mendel, in 1866, established the first rules about phenotypic inheritance. In particular he established that some phenotypic traits are encapsulated in a physical “thing” he called allele and that can be transferred from a parent to its offspring. Then, the discovery of DNA structure gave more support to this idea and we began to call these things “genes”. This leads to a subtle semantic switch, *genetic* inheritance progressively becoming the central dogma to explain any kind of phenotypic inheritance.

Although more and more evidence has accumulated against this dogma (Danchin et al., 2011), the notion of non-genetic inheritance (*i.e.*, any kind of phenotypic inheritance which is not due to genetic inheritance) was considered as playing a minor role in the evolutionary dynamics of species until the work of Jablonka et al. (1995). Since then, the precise role of non-genetic inheritance in the evolutionary dynamics of species has been discussed extensively, in particular in the context of “epigenetic inheritance” which is a subset of non-genetic inheritance implying the physical transfer of molecular states from the parents to their offspring.

On the one hand, epigenetic inheritance could increase the fitness of the organisms that are able to transmit some molecular structures to their descendence. The main argument that backs this point of view is the fact that epigenetic inheritance allows for the transmission of an acquired phenotype. It can thus increase the mean fitness in case of rapid environmental change by eliminating the lag-time of the plastic response (Jablonka et al., 1995; Lachmann and Jablonka, 1996; Bonduriansky and Day, 2009). Moreover, as epigenetic inheritance provides heritable phenotypic variation, it creates the possibility to exploit “Stochastic Gene Expression” to survive and maybe adapt to a hostile environmental variation. Indeed, epigenetic inheritance can help transmitting random phenotypic variations that may be less sensitive to this specific environment and thus promote the evolution of new niche-exploitation strategies (Bonduriansky and Day, 2009). Such a behavior has been observed by Adam et al. (2008): some *E. coli* strains become randomly resistant to an antibiotic and can survive and reproduce in this new environment by partially transmitting this resistance trait to their offspring. Finally, epigenetic inheritance allows to increase phenotypic variation by separating the effect of selection on the genotypes and the phenotypes. Indeed, increasing genetic variability could be dangerous because mutations are mostly deleterious and mutations are not reversible. Epigenetic inheritance enables transmission of the adaptive variations while keeping the mutational burden affordable. This can lead to a new strategy for species: when maladapted to an environment they can increase the stochasticity of their gene expression in order to increase their phenotypic variations. Doing so, they increase the probability of finding an adapted phenotype while still being able to transmit it vertically. This could then facilitate evolution toward a new optimum (Pál, 1998; Pál and Hurst, 2004).

This effect could be enhanced by the ontogenesis in multicellular organisms but this is out of scope of our work.

On the other hand, epigenetic inheritance could decrease the rate of evolution. There are two main points supporting this idea: (*i.*) If the parent environment and the offspring environment are poorly correlated, non-genetic inheritance can be maladaptive. Indeed, the adaptive plastic response of the parent would interfere with the adaptive plastic response of the offspring in the new environment (Bonduriansky and Day, 2009). (*ii.*) Non-genetic inheritance is often transmitted with a probability k different of 0.5. This can alter the fair mendelian game and lead to either a maladaptive variation invading the population because of $k > 0.5$ or an adaptive variation lost because of $k < 0.5$ (Pál and Hurst, 2004).

Unfortunately, there are only few experimental claims in this debate. Most of these assumptions are based on thought experiments and cannot be experimentally studied. Actually, most of the authors still wonder whether epigenetic inheritance really plays a significant role in evolution on the long term. For instance, according to Pál and Hurst (2004): *"Although the role of epigenetic inheritance during development is generally accepted, it is much less clear whether heritable epigenetic variation can play a significant role during evolution"*.

To experimentally address this issue, we propose to use an *in silico* experimental evolution approach. By using an artificial life model, we are able to simulate evolution with and without epigenetic inheritance and to compare the outcomes. To this aim we used the RAevol model, developed in our team (Beslon et al., 2010a,b) to study the evolution of genetic regulation networks. We modified RAevol to allow for the vertical transmission of proteins from a parent to its offspring. In other words, in addition to transmitting its regulation network to its offspring (statically encoded on its genome), a parent also transmits the initial state of this network. Note that this represents only one of the various epigenetic inheritance mechanisms that exist in Nature (e.g. DNA methylation, chromatin structure, protein misfolding...). Furthermore, transmission of proteins is a weak epigenetic transmission because it is easily reversible in case of a plastic response. Nevertheless we show that this mechanisms is strong enough to observe a significant change in the evolutionary dynamics.

This paper is organized as follows: In the next sections, we briefly present the RAevol platform and the way epigenetic inheritance is implemented. We then present our experimental design followed by the results of the simulations which are then discussed. Finally, we conclude and describe our future research directions.

The Aevol - RAevol platform

Aevol is an *in silico* experimental evolution (Hindré et al., 2012; Batut et al., 2013) or digital genetics (Adami, 2006) platform. It was designed to study how the evolutionary conditions shape the molecular structure of an evolving organism (*e.g.*, DNA length, genes number, operonic structures...) due to direct and indirect selective pressures. In Aevol, a population of individuals evolves through a classical mutation-selection process. The specificity of Aevol lies in the genotype-to-phenotype mapping which finely models what is observed in bacteria. A circular double-stranded DNA sequence is transcribed into a set of mRNAs. These mRNAs are then parsed in search for Coding DNA Sequences (CDSs – the "genes") that are translated into proteins through an artificial genetic code. Finally, the proteins are combined to compute the individual's phenotype. Since it is technically impossible to simulate a realistic cellular model, we use a simplified representation. In Aevol the phenotype of an organism is an $[0;1] \rightarrow [0;1]$ mathematical function that represents its metabolism. The fitness is then directly computed by comparing the phenotype with a target function representing the environment. In computational words, Aevol organisms must fulfill a curve-fitting task by combining kernel functions (the proteins) translated from their genomes.

Aevol has been extensively described elsewhere and we refer the reader to previously published work for a complete description of the model and the results obtained so far (Knibbe et al., 2007, 2008; Parsons et al., 2010; Batut et al., 2013; Misevic et al., 2015).

RAevol is an extension of Aevol (Beslon et al., 2010a,b). It uses the same genome model and the same genetic code. However, in RAevol, proteins are able to act as transcription factors (TFs) beside their metabolic activity. When acting as a TF, a protein may up- or down-regulate the transcription of other genes, ultimately controlling the concentration of the proteins encoded by these genes. In other words, in RAevol, each individual owns a genetic regulation network that may dynamically modify its phenotype depending on the environmental conditions. Importantly, in RAevol, the phenotype of an organism is no longer a static function (as it is in Aevol). Rather, it becomes a dynamic function that can be evaluated (by comparing it to a target function) at different time steps during what can now be considered as the "life" of the individual.

Technically, RAevol extends Aevol by adding a "transcriptional regulatory code". In the model, the secondary structure of the proteins (*i.e.*, their sequence of Amino-Acids) may contain small motifs that are able to virtually bind on DNA sequences with an affinity that depends on the matching between the motif and the DNA sequence[1]. If a protein is able to bind upstream or downstream from an mRNA promoter, then it respectively enhances or represses it. This results in an up- or down-regulation of the genes transcribed on this mRNA. Once the regulation network has been decoded from the genome, the transcription levels are

[1] More precisely, the affinity is computed thanks to a regulatory matrix that associates 4AA motifs with 20 nucleotides-long sequences. This matrix is randomly drawn at the beginning of a simulation. Here the same matrix is used for all simulations.

used to parametrize a set of synthesis-degradation equations (one for each protein) and the dynamic of the network can be simulated step by step, enabling the computation of the dynamic phenotypic function during the life of the individual. In RAevol, the target phenotype may change during the life of the individual, either deterministically or randomly (Figure 1). The individual must then dynamically adapt to the current target by switching between different stable states of its regulation network. To this aim, evolution should optimize the "epigenetic landscape" encoded by the regulation network and create pathways between the different local minima associated to the target phenotypes.

All the details of the RAevol model can be found in (Beslon et al., 2010a,b). Classically, in RAevol, when an individual is created, all the proteins' concentrations are initialized at their basal levels (*i.e.*, the level given by their promoter in the absence of regulation). For the needs of the present experiments, we have introduced a new option in the model: protein inheritance. When this option is selected, any new organism created in RAevol inherits its ancestor's genome (possibly mutated) but it also inherits its ancestor's proteins (*i.e.*, the Amino-Acid sequences and the corresponding concentration values). Conversely, all the proteins encoded in the organism's genome are initialized at zero concentration. Then, due to the synthesis-degradation process, the inherited proteins are progressively degraded while the translated ones reach their steady state (see Figure 2, top-right panel). If a given CDS has been correctly transmitted from the parent to the offspring (*i.e.*, its coding sequence has not been altered by a mutation or a chromosomal rearrangement), this process results in a constant activity since the inherited protein is exactly replaced by the produced one. However, when a CDS undergoes a mutation, the offspring inherits a protein that it is not itself able to synthesize and produces a protein its parent was not able to produce. This may interfere with the regulation network convergence, hence altering the individual's phenotype. As a consequence, in case of protein inheritance, the fitness of an organism is not only determined by its genome but also by its inherited cytoplasmic material. Both inheritance processes may thus interfere and we can study experimentally how this interference can impact evolution.

Experimental design

As argued in (Pál, 1998; Jablonka et al., 1995), the dynamic of the environment can be a crucial parameter for the contribution of epigenetic inheritance to evolution. Thus, in order to test whether epigenetic inheritance influences the evolutionary dynamics, we designed 2 series of scenarios. One will test the effect of protein inheritance in a variable environment and the other, in a constant environment.

In both scenarios the individuals live for 20 time steps (Figures 1 and 2). However, in the constant environment, the environmental conditions are fixed along the whole evolutionary process. More precisely, the phenotypic target is the sum of four Gaussians regularly spread along the phenotypic axis. On the opposite, in the case of a variable environment, we designed three different environmental conditions that alternate randomly with a switching probability of 10% at each time step (including the first one). In each of these conditions, one of the four Gaussians is active while the three others are inactive (Figure 1). In such an environment individuals can adapt by plasticity since the same conditions are encountered regularly (though randomly). Moreover, since the birth condition of an individual has a 0.9 probability to be its parental one, protein inheritance may reduce the lag time of the plasticity response. We will thus be able to test the theory proposed by Jablonka et al. (1995), to evaluate its contribution to evolution and to seek for additional mechanisms.

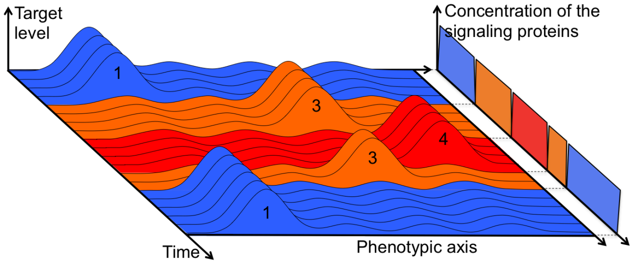


Figure 1: Example of a variable environment. The three environmental conditions are displayed respectively in blue, orange and red. They alternate randomly with a switching probability of 10% at each time step. Individuals life-duration is constant, fixed to 20 time steps. Each condition is also characterized by a signaling protein that can trigger a plastic response.

To summarize, we have four situations to simulate: variable environment with protein inheritance (V_i), variable environment without protein inheritance (V, control for V_i scenarios), constant environment with protein inheritance (C_i) and constant environment without protein inheritance (C, control for C_i scenarios).

For each scenario, we launched 4 independent simulations. In each simulation, a population of 1,000 individuals was let evolve for 300,000 generations under a mild mutation rate. All parameters were the same for all the simulations and were chosen according to previous experiments (Mutation rates: 5.10^{-6} mut/bp/generation for all kinds of point mutations. Rearrangement rates: 5.10^{-5} rearr/bp/generation for all kinds of rearrangements. Fitness proportionate selection. Selection coefficient: 750). Without epigenetic inheritance these parameters were known to lead to a medium complexity organisms with a good final fitness. In particular, they always lead to the emergence of

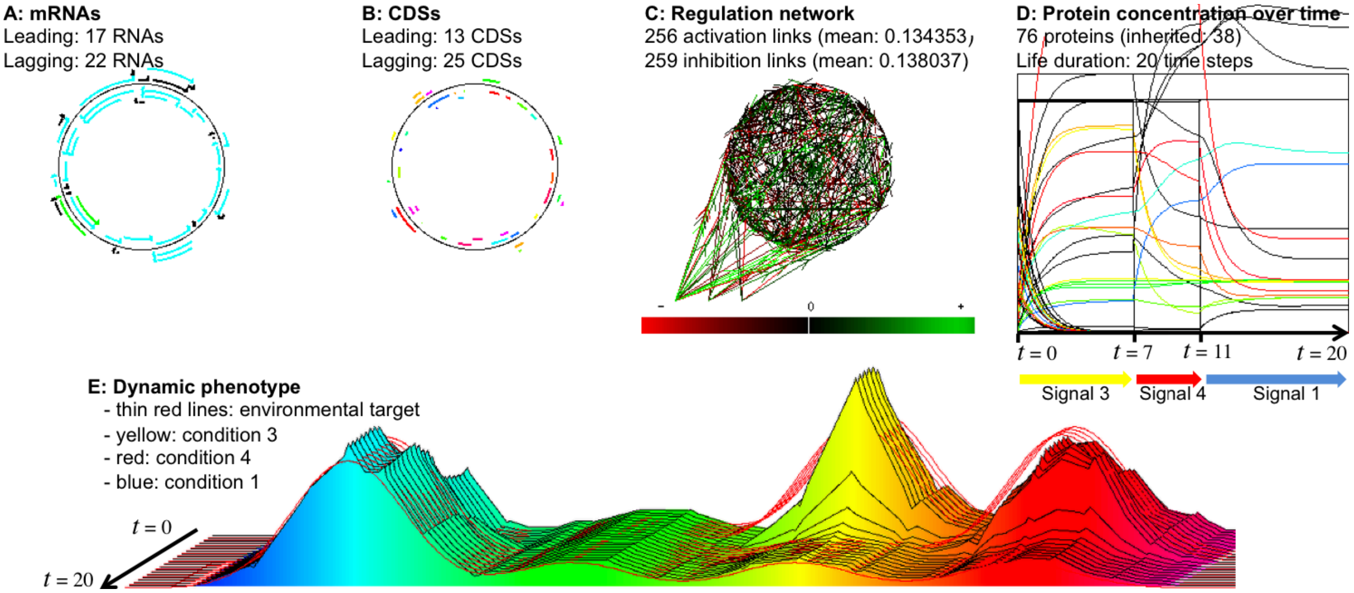


Figure 2: Example of an evolved Raevol individual after 300,000 generations in V_i conditions (variable environment and epigenetic inheritance). **A**: genome and mRNAs (colors code for the basal transcription rate); **B**: genome and CDSs (colors indicate the function of the gene product); **C**: Regulation network (arrows indicate links between genes and mRNAs; colors indicate the weight of the link. Note the three external signals displayed at the bottom-left of the network); **D**: protein concentrations during the life of the individual (colors indicate the function of the protein). **E**: Dynamic phenotype of the individual (same color code as panels **B** and **D**; thin red lines indicate the target at each time step). This individual is born in condition 3. At time 7 the environment switches to condition 4 and at time 11 it switches to condition 1 (see Figure 1 for a detailed explanation). At each environmental change the regulation network modifies the transcription levels and thence protein concentrations, resulting in a phenotypic adaptation. Protein inheritance is clearly visible on panel **D**: 38 proteins are inherited (initialized at concentrations inherited from the ancestor but rapidly dropping to zero) and 38 are produced during the individual's life (initialized at concentration zero but rapidly reaching their steady-state).

a genetic regulation network as early as 50,000 generations. These parameters (and the relatively low number of repetitions) were chosen as a compromise between long enough evolution time and short enough computation time. Indeed, these simulations already last for more than 1 month on a 16-core 3GHz computer. It is important to mention that there is still room for directional evolution after 300,000 generations but previous experiments have shown that the main tendencies are already observed at this stage with the used parameter set. Indeed, during preliminary tests, we ran similar simulations for 800,000 generations without observing dramatic changes in the last period.

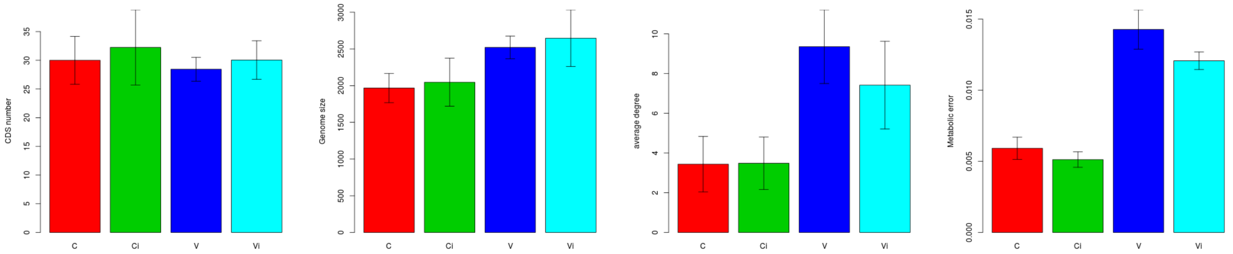
Results

All 16 simulations resulted in the emergence of organisms of medium complexity with most genomes ranging from 2,000 to 3,000 bp and approximately 30 genes (Figures 3.a and 3.b). Moreover, all the individuals were able to regulate their transcription and to efficiently respond to the environment changes (in the case of scenarios V/V_i). Figures 2 and 4 show the best final V_i individual and its genetic regulation network. Note the very fast proteome reorganization after each environment variation (Figure 2, top-right panel).

As RAevol is a precise model of genome evolution, we are able to check whether protein inheritance has a major impact on the genome structure after 300,000 generations. As Figure 3 shows, there is no significant difference in neither the genome structure nor the genetic regulation network between scenarios with or without protein inheritance.

Conversely it is interesting to note that we do observe differences between scenarios C/C_i and V/V_i for the genome size (Figure 3.b) and average degree of the regulation network (Figure 3.c) but not for the number of genes (Figure 3.a). This strikingly contrasts with the classical hypothesis (see (Casjens, 1998) for example) that postulates that a link exists between the complexity of the environment and the size of the regulation network.

Here, the complexity of the environment is clearly higher for scenarios V/V_i than for scenarios C/C_i. However, the sizes of the regulation networks are similar (Figure 3.a) although the connectivity is lower for scenarios C/C_i (Figure 3.c). Moreover, the difference in molecular complexity observed here are orders of magnitude lower than what was previously observed in the same model when varying



(a) Number of CDSs (b) Genome size (c) Average degree (d) Metabolic error

Figure 3: Comparison of the gene number, genome size, average degree of the regulation network and metabolic error of the best individual for the 4 scenarios at generation 300,000 (mean values of the 1000 last generations). Error bars show the standard deviation between the 4 simulations of a scenario. Red: scenario C, green: C_i, blue: V, cyan: V_i.

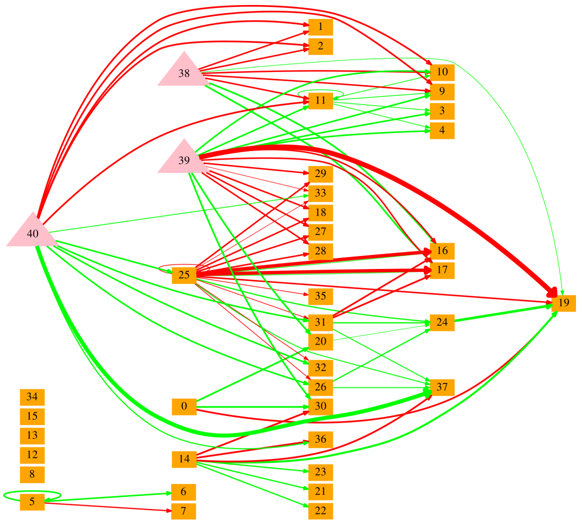


Figure 4: Genetic regulation network of the best individual in scenario V_i. Orange boxes represent genes; pink triangles represent the 3 signal proteins. Red (resp. green) arrows represent inhibition (resp. activation) links. Arrows weight represents the intensity of the link. For the sake of clarity, only links whose weight are at least 33% of that of the strongest link are represented.

the mutation rates rather than the environmental complexity (Beslon et al., 2010b). This observation raises very interesting questions regarding the origin of molecular complexity.

Although no clear molecular difference was observed between the protein inheritance scenarios (C_i/V_i) and their control (C/V), Figure 3.d shows a slight difference between them in terms of the individuals' performances. Indeed, the comparison of the mean metabolic error[2] of each scenario shows us that scenarios V_i and C_i have lower metabolic error than scenarios V and C respectively (note that statistical accuracy cannot be reached with only 4 repetitions).

To understand this result, one can look at the evolution of metabolic error over time. Figure 5 clearly shows that protein inheritance accelerates the decrease of the metabolic error during the second phase of evolution (from 150,000 to 300,000 generations). However when looking qualitatively at the curves, we can notice that the evolutionary dynamic is clearly different with or without protein inheritance. In particular scenario C_i is characterized by long stasis phases separated by avalanches of favorable mutations, a situation that is not observed for scenario C. On the opposite, scenario V_i shows large fluctuations of metabolic error during the first phase of evolution (generation 0 to 50,000). These fluctuations initially slow-down evolution but scenario V_i ultimately catch up with scenario V because of a more regular evolution from generations 50,000 to 300,000. We can thus propose the hypothesis that the effects of protein inheritance are different depending on the evolutionary phase. Indeed, it seems to have a negative impact during the first evolution period (*i.e.*, when the network is being constructed by node recruitment) while its impact is clearly beneficial in the second period (*i.e.*, when the network structure is mostly stable but when the links are being optimized).

This hypothesis is confirmed when observing the evolution of the mean relative advantage given by protein inheritance over time. Indeed, Figure 6 confirms that the impact of protein inheritance is different in a constant environment and in a variable environment and that protein inheritance influence shows strong variations along the evolutionary process. For scenarios C/C_i, the advantage is slightly negative until generation 200,000 but with large fluctuations. Then it becomes stable and positive at almost 14%. This behavior is a direct consequence of the long evolutionary stasis phases

[2] In Aevol/RAevol the adaptation of an individual is measured by its "metabolic error" *i.e.*, the total difference between its phenotypic function and the target function. Note that the lower the metabolic error, the better the individual (contrary to the usual fitness measures).

previously observed for scenario C_i (Figure 5). Moreover these phases are separated by large drops in the metabolic errors. Thus the relative advantage/disadvantage of protein inheritance is driven by stochastic events (the beneficial mutation avalanches). Given the low number of repetitions, this results in the large fluctuations observed on figure 6..

For scenarios V/V_i, we observe a huge disadvantage of protein inheritance that reaches -40% at generation 20,000. However this is rapidly compensated and protein inheritance becomes favorable as soon as generation 70,000. This is due to large fluctuations in the metabolic error in two repetitions of scenario V_i (Figure 5). Once these fluctuations stop, all the simulations rapidly reach a higher fitness than in scenario V. Interestingly, similar fluctuations are observed for the evolution of the number of CDSs (Figure 7). This confirms that the negative impact of protein inheritance is linked to the process of node recruitment by the regulation network.

As shown previously, during the second phase of the evolutionary process, protein inheritance seems to favor evolution whatever the conditions. However the advantage it provides can be due either to some facilitating process that would accelerate evolution or to a more “mechanical” effect: As explained above, protein inheritance can directly decrease the metabolic error because it enables transmission of an acquired phenotype. This is likely to be favorable when there is a high probability that an offspring faces the same conditions as its parents. In our simulations, this effect is likely to play an important role since the probability for an individual to begin its life facing the same environment its parent ended its life with is 90% in scenarios V/V_i and, obviously, 100% in scenarios C/C_i.

In order to evaluate the contribution of this effect, we measured the proportion of metabolic error that is due to the delay between an individual initialization and the moment when its regulation network reaches its steady state, for all the possible environmental conditions. At generation 300,000 we found a relative advantage of 7.5% (resp. 0.5%) for scenario V_i (resp C_i) over scenario V (resp C). Note that our estimation process overestimates the contribution of this effect since it does not take into account the possibility of an environmental switch immediately before or after the individual initialization (*i.e.*, the possibility that the environmental conditions change before the network reaches its steady state). As the mean relative metabolic error gain due to protein inheritance was 16,7% for scenarios V_i/V and 14.4% for scenarios C_i/C at generation 300,000, we can conclude that the final advantage observed for scenarios C_i and V_i is a combined effect of the direct transmission of an efficient acquired phenotype and of another yet to be discovered “facilitating” effect that accelerates fitness gain.

Discussion

We have presented two main results of our experiments. First, protein inheritance increases the long term fitness gain during evolution for scenarios C/C_i and V/V_i. Second, it has a negative effect during the first phase of the simulations: it generates instability in scenario V_i and clogging before fitness improvement for scenario C_i[3]. In this section, we propose hypotheses to explain both observations.

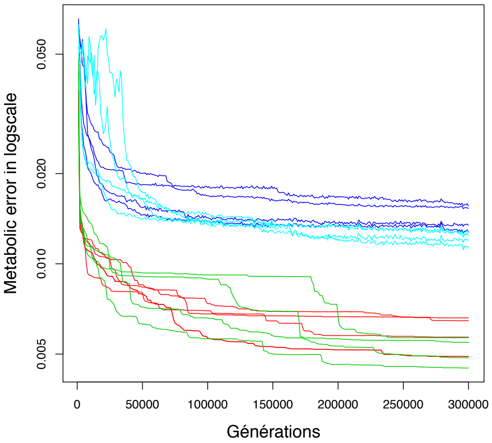


Figure 5: Evolution of the metabolic error of the best individual in scenarios C (red), C_i (green), V (blue) and V_i (cyan). For scenarios V and V_i the environmental condition changes randomly. This create random variation of the metabolic error between generations even for a same individual. To avoid this effect each point of the curve is the mean value of the 1,000 previous generations.

Protein inheritance leads to major evolution impediment in the first phase of the simulations (*i.e.*, during the phase of gene acquisition and regulation network construction). This can be explained by a dynamic conflict between the two forms of inheritance that interact in the model. Indeed, the expression of a mutation does not follow the same timing for genetic inheritance and for protein inheritance. Indeed, if a non-silent mutation occurs at generation n, it leads to the emergence of a new phenotype at generation $n+1$. However this new genotype will be expressed in the epigenetic context transmitted from generation n (*i.e.*, in an epigenetic context that does not include the mutation). Things are different at generation $n+2$ where both the genetic inheritance and the epigenetic inheritance transmit the mutation. Depending on the genetic background and on the kind of mutation, this effect can be almost neutral but it can also have important consequences if the genetic and epigenetic contributions of the mutation are of opposed sign. A mutation with a positive genetic effect and a negative epigenetic effect could be beneficial at generation $n+1$ (and thus be selected for) while being deleterious at generation $n+2$. Similarly, a mutation

[3] The scenarios C and C_i can be compared by measuring the standard deviation of the derivative of the fitness. We find that the standard deviation is twice higher in scenario C_i than in scenario C (3.7710^{-7} vs. 1.6310^{-7}).

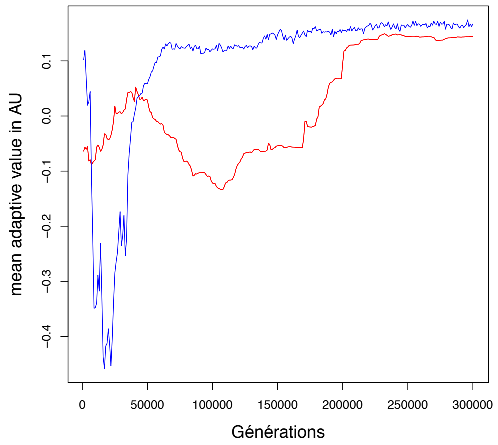


Figure 6: Evolution of the relative advantage of the protein inheritance in scenarios V/V_i (blue) and C/C_i (red). The relative advantage is computed as the difference of metabolic error divided by the mean metabolic error between the scenarios with and without protein inheritance.

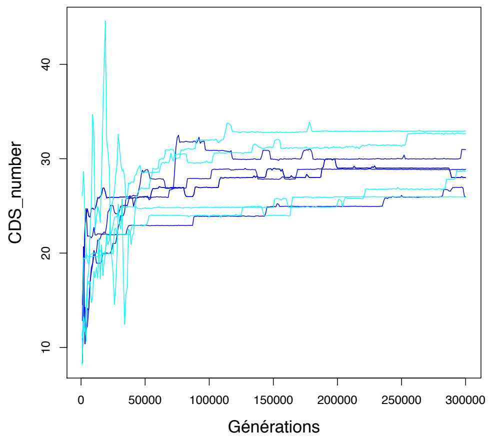


Figure 7: Evolution of the number of genes of the best individual in scenario V (blue) and V_i (cyan). As for figure 5, the curves have been smoothed.

with a negative genetic effect and a positive epigenetic effect could be counter-selected at generation $n+1$ while it would have been favorable later.

This mechanism is likely to have a different impact on scenario V_i and C_i. Indeed, for scenario C_i, the target phenotype is constant and the genetic regulation network is not mandatory to survive (although it may be useful to finely tune the protein concentrations). Consequently, the interference between the two modes of selection adds noise to the evaluation function and may impede the fixation of favorable mutations but it is not severely detrimental. On the opposite, for scenario V_i, the regulation network is mandatory to survive in the three conditions that alternate in the environment. Moreover, evolution must create some pathways in the epigenetic landscape for the individuals to be able to switch from one target to another. There is thus a permanent competition between two evolutionary strategies: evolving a complex network that is able to efficiently switch between the environmental conditions, or keeping the population polymorphic (*i.e.*, to maintain subpopulations that only fit one of the environmental conditions). Given the switching probability, the regulation strategy is much more efficient but it is also much more affected by the interference between genetic inheritance and epigenetic inheritance since an efficient regulation network may be counter-selected “simply” because it is not initialized in the correct basin of attraction. Ultimately, this delays the moment when the regulation strategy invades the population, resulting in large fluctuations of the metabolic error.

Interestingly, this effect is also likely to depend on the kind of mutations. Indeed, if a mutation only affects an existing regulation link or an existing CDS, the consequences on the epigenetic inheritance are likely to be mild (since there is only quantitative variations). On the opposite, if the mutation corresponds to the creation (respectively deletion) of a gene, this can have dramatic consequences since the mutation will add (respectively remove) a protein at generation $n+1$. In such conditions, the interference between genetic and epigenetic inheritance is maximal. Indeed, a simple comparison between Figures 5 and 7 clearly shows that the fluctuations of metabolic error occur simultaneously with large fluctuations of the number of genes. This could explain why epigenetic inheritance is negative in the first evolutionary phase (when many genes are acquired in a “few” generations) but positive in the second phase.

The positive effect of epigenetic inheritance on the long term is much more straightforward. First, as we have seen above, the advantage of epigenetic inheritance must be decomposed between a direct advantage (the transmission of the epigenetic memory reduces the lag phase when the environmental conditions of the parent and of the offspring are correlated – which is often the case in our simulation) and an indirect advantage that accelerates evolution. This indirect advantage is easily explained in terms of pathways in the epigenetic landscape. Indeed, in scenario V_i, the epigenetic landscape must contain 3 pathways to enable switching between the different environmental conditions: $1 \leftrightarrow 2$, $1 \leftrightarrow 3$ and $2 \leftrightarrow 3$ (each of them triggered by a signaling protein). However, in scenario V, the epigenetic landscape must contain 6 pathways: $1 \leftrightarrow 2$, $1 \leftrightarrow 3$, $2 \leftrightarrow 3$ *plus* $I \rightarrow 1$, $I \rightarrow 2$ and $I \rightarrow 3$ (I being the initial state of the regulation network). Consequently, the genetic network is much more constrained for the individuals in scenario V than for those in scenario V_i. The situation is similar for individuals in scenario C that must evolve an epigenetic pathway from their initial state to their steady state while in scenario C_i, individuals are always initialized at the steady state.

Conclusion and future work

In this paper, we experimentally addressed the question of the impact of non-genetic inheritance on evolution (*i.e.*, whether it could accelerate or decrease the rate of evolution). To evaluate this question, we used an *in-silico* experimental evolution platform, RAevol, to test 4 scenarios: 2 with a static environment (C and C_i) and 2 with a dynamic one (V and V_i). For each, we launch 2 types of simulations: 2 with protein inheritance (C_i and V_i) and 2 without (C and V). Our simulations show that protein inheritance indeed increases the rate of evolution on the long term but that it can have a strong negative effect at the beginning of the evolutionary process by generating instability in V_i scenario and clogging before fitness jumps for scenario C_i. Finally, we discussed these results and proposed that the evolutionary consequences of protein inheritance are a complex composition of three mechanisms: (*i.*) interference between genetic and protein inheritance due to their different transmission delays. This creates noise in the selection process and may delay the emergence of an efficient regulation network; (*ii.*) epigenetic memory that reduces the lag time when parents and offspring environments are correlated and (*iii.*) simplification of the epigenetic landscape.

The former effect was shown to be strongly deleterious in our simulations but only at the beginning of the evolution. It is then rapidly compensated by the two latter, resulting in a long-term benefit of protein inheritance on the evolution of our artificial organisms.

To the best of our knowledge, this work is the first attempt to explore experimentally the question of the impact of epigenetic inheritance on evolutionary dynamic. We now need to validate statistically our results by running more experimentation and to test other environmental conditions to confirm our mechanistic hypothesis (*e.g.*, by letting individuals evolve in an environment where variation is correlated with the reproduction).

Availability

Aevol is available under GPL licencing at the project website: http://www.aevol.fr. RAevol is currently in beta-version and is available upon request from the authors.

Acknowledgements

We would like to warmly thank David P. Parsons for careful reading of the manuscript. This research has been supported by EU-FET grant EvoEvo (ICT-610427).

References

Adam, M., Murali, B., Glenn, N., and Potter, S. (2008). Epigenetic inheritance based evolution of antibiotic resistance in bacteria. *BMC Evolutionary Biology*, 8:12.

Adami, C. (2006). Digital genetics: unravelling the genetic basis of evolution. *Nature Reviews Genetics*, 7(2):109–118.

Batut, B., Parsons, D. P., Fischer, S., Beslon, G., and Knibbe, C. (2013). In silico experimental evolution: a tool to test evolutionary scenarios. *BMC Bioinformatics*, 14 (S15):S11.

Beslon, G., Parsons, D. P., Peña, J.-M., Rigotti, C., and Sanchez-Dehesa, Y. (2010a). From digital genetics to knowledge discovery: Perspectives in genetic network understanding. *Intelligent Data Analysis Journal*, 14(2):173–191.

Beslon, G., Parsons, D. P., Sanchez-Dehesa, Y., Peña, J.-M., and Knibbe, C. (2010b). Scaling laws in bacterial genomes: A side-effect of selection of mutational robustness. *BioSystems*, 102(1):32–40.

Bonduriansky, R. and Day, T. (2009). Nongenetic inheritance and its evolutionary implications. *Annual Review of Ecology, Evolution, and Systematics*, 40:103–125.

Casjens, S. (1998). The diverse and dynamic structure of bacterial genomes. *Annual review of genetics*, 32(1):339–377.

Danchin, E., Charmantier, A., Champagne, F. A., Mesoudi, A., Pujol, B., and Blanchet, S. (2011). Beyond DNA: integrating inclusive inheritance into an extended theory of evolution. *Nature Reviews Genetics*, 12(7):475–486.

Hindré, T., Knibbe, C., Beslon, G., and Schneider, D. (2012). New insights into bacterial adaptation through in vivo and in silico experimental evolution. *Nature Reviews Microbiology*, 10:352–365.

Jablonka, E., Oborny, B., Molnar, I., Kisdi, E., Hofbauer, J., and Czaran, T. (1995). The adaptive advantage of phenotypic memory in changing environments. *Philosophical Transactions of the Royal Society B*, 350:133–141.

Knibbe, C., Coulon, A., Mazet, O., Fayard, J.-M., and Beslon, G. (2007). A long-term evolutionary pressure on the amount of noncoding DNA. *Molecular Biology and Evolution*, 24(10):2344–2353.

Knibbe, C., Fayard, J.-M., and Beslon, G. (2008). The topology of the protein network influences the dynamics of gene order: From systems biology to a systemic understanding of evolution. *Artificial Life*, 14(1):149–156.

Lachmann, M. and Jablonka, E. (1996). The inheritance of phenotypes: an adaptation to fluctuating environments. *Journal of Theoretical Biology*, 181:1–9.

Misevic, D., Frénoy, A., Lindner, A. B., and Taddei, F. (2015). Shape matters: Lifecycle of cooperative patches promotes cooperation in bulky populations. *Evolution*, 69(3):788–802.

Pál, C. (1998). Plasticity, memory and the adaptive landscape of the genotype. *Proceedings of the Royal Society of London B: Biological Sciences*, 265:1319–1323.

Pál, C. and Hurst, L. D. (2004). Epigenetic inheritance and evolutionary adaptation. In *Organelles, Genomes and Eukaryote Phylogeny: An Evolutionary Synthesis in the Age of Genomics*. CRCpress.

Parsons, D. P., Knibbe, C., and Beslon, G. (2010). Importance of the rearrangement rates on the organization of transcription. In *Proceedings of Artificial Life XII*, pages 479–486.

Probing models of minimal swimming vehicules in vivo with microalgae phototaxis.

David Colliaux[1], Lia Giraud[2], Claude Yéprémian[3], Pierre Bessière[1] and Jacques Droulez[1]

[1]ISIR, UMR7222 CNRS/Paris-Sorbonne University, Paris
[2] SACRe Ph.D candidate, PSL Research University, EnsadLab and LCLCP-UMR 7574, Paris
[3] CCE, UMR 7245 CNRS/Muséum National d'Histoire Naturelle, Paris
david.colliaux@isir.upmc.fr

Abstract

We divert an experimental device from its artistic purpose to test the properties of minimal sensorimotor loops resulting in phototactic behaviour. The study of microalgae like Euglena or Chlamydomonas is coupled with a modeling effort to characterize the swimming behaviour.

The microalgae *Chlamydomonas Reinhardtii* has been proven to be a useful model organism for photosynthesis or cilia physiology. Based on recent studies of its swimming behaviour from Goldstein (2015), experimental measurement of phototaxis and mathematical model for the dynamics of the cell density, we propose that it also provides a framework to study sensorimotor loops. The ability of Chlamydomonas to move toward or to escape from a source is a minimal biological example of *vehicule* behaviour as in Braitenberg (1986).

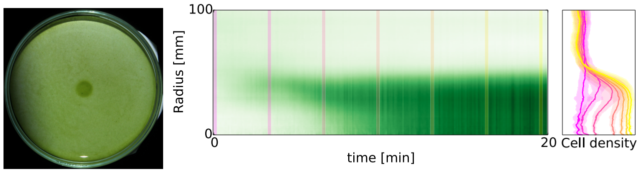


Figure 1: Phototactic behaviour: **Left** Picture of the Petri dish after 20 minutes illumination by a centered disk of light. **Middle** Time course of the cell density as relected by the pixel intensity depending on the distance from the center. **Right** Cell density as a function of the radius (± std) at various times marked on the time course.

Macroscopic measures of phototaxis.

The illumination of a culture of Chlamydomonas in a Petri dish generates changes in the density of cells. In the example of figure 1, the moderate light stimulation at the center (5mm disk) results in accumulation of the cell at the lightened area. With the time lapse record of the Petri dish (12fps), it is possible to characterize the dynamics of phototaxis and to track the spatial profile of cell density as reflected by the pixel intensity (not that the decrease in cell density outside of the lightened spot is barely visible).

Microscopic mechanisms and cell density dynamics.

Intracellular fluctuations modulated by the light intensity generate reorientations of the helical swimming trajectory of the cell by desynchronization of the 2 flagella at random times with a higher rate when the cell is far from the light source. The microscopic components and simulations snapshots of the resulting biased random walk are illustrated on figure 2.

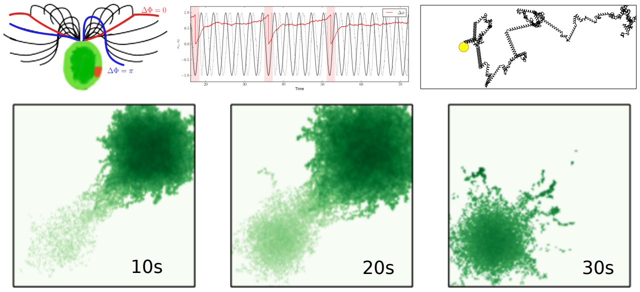


Figure 2: **Up-left** Flagella configurations for synchronized (red) and desynchronized state (blue). **Up-middle** Simulation of stochastic coupled oscillators (Adler model similar to Goldstein (2015)). **Up-right** Sample trajectory of an algae moving to the light source (yellow spot), combining run and tumble trajectory and helical motion. **Bottom** Density of a population initialized with Gaussian distribution moving toward a light source.

Interacting with a microalgae population is thus a way to characterize its sensorimotor loop.

References

Braitenberg, V. (1986). *Vehicles: Experiments in Synthetic Psychology*. MIT Press.

Goldstein, R. E. (2015). Green algae as model organisms for biological fluid dynamics. *Annual Review of Fluid Mechanics*, 47(1):343–375.

Towards Failure-Resistant Mobile Distributed Systems Inspired by Swarm Intelligence and Trophallaxis

Arles Rodríguez[1], Jonatan Gómez[1] and Ada Diaconescu[2]

[1]ALIFE Research Group, Universidad Nacional de Colombia
[2]Telecom ParisTech, LTCI CNRS
{aerodriguezp,jgomezpe}@unal.edu.co, ada.diaconescu@telecom-paristech.fr

Abstract

Designing distributed algorithms for mobile ad-hoc sensor systems is difficult, not at least because of their asynchronous communication, mobility, absence of shared memory and high risk of failures. To deal with these challenges, some techniques like replication and consensus are proposed in the literature. However, techniques like consensus depend on a leader election, and this leader can fail. In this paper, we present some advances inspired from nature for the design of a decentralized, scalable way for getting and synchronizing information among components in a point-to-point way. To achieve this, we address the problem of getting and synchronizing information by defining a Distributed System as a swarm of agents (termites), which look for information. Termites are designed with the task of exploring a simulated environment, sensing some desired data distributed throughout the space, and sharing their local knowledge regarding the environment with other nestmates only if they are neighbors. However, when failure rates increase it is less probable than a termite completes the entire task by itself before all termites fail. In order to allow at least one termite to gather the complete information from the environment, several solution approaches are proposed, like sequential exploration with one agent as reference, random movements with local information exchanges, Levy walks and a pheromone-based exploration algorithm inspired by Ant Colony System. This algorithm allows a termite to explore data in a world and enables a termite to search other nestmates with more information by using a trace and by defining a search status given the amount of local information than a termite has. Results show, how swarms manage to collect and replicate information from the entire space even when failures occur. By local interactions, almost all the termites get complete information from a defined world before failing, without a central control and with simple local rules.

Introduction

A Distributed System consists of a collection of components (e.g. processes, agents, robots and nodes) connected via a network, that coordinate their activities and share system resources, offering different services to users and appearing to them as a single system (Tanenbaum and Steen, 2006). Coordination and cooperation between processes are necessary and communication is a key point of Distributed Systems from a design point of view because each process has partial knowledge of the environment, acts in a local way and can fail (Raynal, 2013).

Distributed systems must be scalable. It means being adaptable to changes in terms of size (to provide an easy way to add or delete resources), geographical localization of components and provide easy management independently of its size (Tanenbaum and Steen, 2006). Scalability involves the design and implementation of decentralized algorithms dealing with components that can fail, have no complete information, take decisions in a local way and use point-to-point communication because broadcast is not possible. Lack of scalability usually implies loss of performance of a system while a system grows up (Tanenbaum and Steen, 2006).

Autonomic Computing addresses complexity with the idea of a computer system that adapts to changes without human intervention (Lalanda et al., 2013). Inspired by nature, the idea is that systems elements manage themselves while also providing their services (Kephart and Chess, 2003). The internal behavior of an autonomic element and the set of relationships with other elements are based on goals that a designer has embedded in it and on goals incorporated by other systems through subcontracts with other elements with its tacit or explicit consent (Kephart and Chess, 2003). A set of Self-* properties are required to achieve from the Self-management goal: adapting to the addition or deletion of components (Self-configuration), detecting and recovering from failures without disruption in the system operation (Self-healing), finding improvements in the efficiency of a system (Self-optimization), and anticipating and preventing of threats (Self-protection) (Lalanda et al., 2013; Kephart and Chess, 2003).

On the other hand, replication is proposed as an essential component of failure tolerance in distributed systems. A well known mechanism of replication in Distributed Systems is consensus. By consensus, it is expected that replicas have agreement over a value given local information in each process (Aguilera, 2010). Different areas of application includes state and log machine replication in databases (Aguilera, 2010; Ongaro and Ousterhout, 2013), motion planning, alignment problems (Nedic et al., 2010) and information processing in sensor networks (computing averages of local observations) (Ozdaglar and Nédic, 2007).

Some existing consensus algorithms like Paxos (Lamport, 1998) and Raph (Ongaro and Ousterhout, 2013) propose distributed consensus algorithms based on a leader election process. This leader receives information and replicates it to other processes. However, natural systems do not require a leader directing them and self-organisation relies on collective behaviors that emerge from local interactions as happens in insect colonies where collaboration emerges from Stigmergy (Doursat et al., 2012).

An important challenge to deal with failures is to collect, replicate and synchronize information in a fast way based on the development of adaptable ways of point-to-point communication. In this paper, we propose a solution to the sharing information problem that deals with some of the challenges introduced before. A simulated world is defined with some $data$ of interest distributed in the environment. To get information in this world, we define agents called termites with the task of exploring and obtaining this desired $data$. A termite can move, sense $data$ in its current location and can share its local information with other neighbors. However, termites are unreliable and can crash with a given probability p_f. Given these conditions, the problem is how to get at least one termite to collect whole information of a world before they all crash.

Agents are called "termites" because they follow social organization and their feeding is carried out via *trophallaxis*, meaning that food is stored in their stomach and it is transferred among nestmates through mouth-to-mouth feeding (Rodriguez and Gomez, 2011). In this paper, local information is also inside each agent and local exchange of information is performed between neighbors that look for more information present in other nestmates using stigmergy. *Trophallaxis* is important in nutritional dynamics and communication of many social insects, individual foragers return from a food resource and transfer a portion of their gut material to one or several nestmates. These recipients subsequently become donors to others, and the process continues (Suárez and Thorne, 2000).

This paper uses stigmergy for guiding termites in their search of new information enabling termites to explore a terrain, to get data information from other nestmates and to synchronize local information with others. Our approach inspired by trophallaxis and swarms allows termites to get the whole data before they crash in a decentralized way. A swarm features self-organization that allows it to continue working even if some termites die looking for new information. Inspired by this feature, by adding failures to termites we aim to determine if our proposal can achieve terrain exploration and failure resistance in a simple way inspired by nature; and also establish limits. The remaining of this paper is organized as follows: next section presents a detailed description of the problem, the following section shows some approaches to solve the problem including our proposal of sharing information based on stigmergy and trophallaxis. Finally a result analysis is performed and some conclusions are drawn.

The Problem: World exploration

A world is a bidimensional toroidal space defined as a matrix of properties $Props^{width \times height}$. $Props$ is a collection $Props = \{\tau_w, data\}$, defined with the following values: amount of pheromone in the world $\tau_w : \tau_w \in \mathbb{R} \wedge \tau_w \in [0, 1]$ and $data : data \in \mathbb{R} \wedge data \in [0, 1]$. $data$ represents some information of interest in this world (e.g. temperatures, altitudes, distance to a determined objective). By now, $data$ values do not change because the main objective is to find a way to get all $data$ information and share it in a fast way. At the beginning all the world positions have a pheromone value of 0.5 and $data$ is generated for each location in a random fashion.

Termites Definition

Multi-agent systems define agents capable of independent actions with the ability of interacting with others. In order to achieve their tasks they are required to cooperate, coordinate and negotiate which each other (Balaji and Srinivasan, 2010). Each termite senses information from the environment and acts by using actuators (Russell and Norvig, 2004). Agents by now are reactive. It means, that each termite operates to respond to changes and to satisfy its design objectives (Russell and Norvig, 2004). Objectives are defined in a termite program which determines the action to be executed by an agent according to its local knowledge. The main thread of each agent looks like algorithm 1. While an agent is alive `(status != Action.DIE)` this agent senses its environment, then chooses an action based on its perceptions and finally the action has an effect on the environment.

```
while (status != Action.DIE) {
    Percept p = environment.sense(this);
    Action action = compute(p);
    environment.act(this, action);
}
```

Algorithm 1: Termite main program

In this paper, each termite is designed with the objective of getting $data$ information in a simulated world of size $width \times height$. $data$ values are represented as a matrix of continuous values $\mathbb{R}^{width \times height}$. The idea is that termites cooperate and coordinate among them to get the global $data$ information starting from local perceptions. The main idea is that each agent looks for and senses new $data$ locally and shares its collected $data^{width \times height}$ at the same time.

Different perceptions have been defined for termites $Percept = \{pheromone, data, socialStatus, neighbor, msg, loc\}$. $pheromone$ is a vector $\mathbb{R}^n$ with values in $[0, 1]$ representing the amount of pheromone that a termite has in its vicinity (Moore neighborhood $r = 1$ with center in the termite location (Gray, 2002)); $data$ is the information in the current location of the termite, $socialStatus = \{\texttt{SEEKER}, \texttt{CARRIER}\}$ indicates the status of a termite, $neighbor$ returns the id of a nestmate randomly selected from its Moore neighborhood if one exists, msg stores new messages received from other nestmates and loc returns the current termite location as a matrix position $(row,\ column)$.

A termite has the following actions $Actions = \{none, down, left, right, up, upleft, upright, downright, downleft, Die, Collect, Send, Receive\}$. First nine actions make reference to movements in the world and Die stops an agent's thread to simulate failure. $Collect$ is performed in each round and means to store the $data$ collected in the local memory of termite. A termite s sends its $data$ collected I_s to a termite r in a msg encoded as a collection $msg = [I_s]$ in a process defined as $Send(r, msg)$. Sending this information is inspired by traditional Asynchronous Distributed Systems where there are FIFO communication channels (Messages received first are processed first by each agent (Raynal, 2013)), local communication (each agent receives and sends messages only to its neighbors), and there is no timing assumptions regarding message delay, clock drift or time taken to send a message. The control returns back to the invoking process after the data is copied in the buffer of the process that receives (Chandra and Toueg, 1996; Kshemkalyani and Singhal, 2008). To implement this mechanism there is a FIFO queue mailbox in the world for each neighbor and an internal queue in each agent that loads new data using the msg perception. In an analogous way, each time a termite r receives new information from a neighbor $Recv(msg)$, it decodes the received message $msg = [I_s]$ and takes decisions based on this information depending on the solution approach.

Trophallaxis inspires the information interchange of this approach. When two termites are occupying adjacent locations in the world, they exchange information. In this way, and just like in nature, termites are donors to other neighbors in a cascade scheme called *Trophallactic cascade*. This pattern of transfer may prove to be more efficient and result in more equitable distribution than direct transfer in nature (Suárez and Thorne, 2000). Communication is added in a way that each time a termite s senses a neighbor r, it gets its id, and exchanges its local $data$ information with r by using $Send$ and $Receive$ in the following way:

- Termite s performs $Send(r, msg)$ where $msg = \{I_s\}$. In this way a termite s sends its current information I_s to r.

- Termite r receives $Recv(msg)$ and completes its information $I_r = I_r \cup I_s$.

Failure Definition

One known type of Distributed System failure is a crash. That is, a process of a Distributed System halts but is working fine until it halts. When a process fails nothing is heard from it. This failure can be identified because the node stops sending messages and does not report a failure (Tanenbaum and Steen, 2006; Satzger, 2008). In the proposed model, a crash is equivalent to the death of a termite. This kind of failure is implemented by defining a probability of failure (p_f) for the agents, see Algorithm 2. For example, $p_f = 0.1$ means that a process has a probability of failure in 1 of 10 rounds.

```
//process can fail with a probability of 0.1
double probFailure = 0.1;
Percept p;
Action actions;

while (status != Action.DIE) {
    if (Math.random() < probFailure) {
           status = Action.DIE;
    }

    p = environment.sense(this);
    actions = compute(p);
    environment.act(this, actions);
}
```

Algorithm 2: Crash model for Agent Program of termite i

Solution Approaches

In this section, some approaches are proposed to solve the aforementioned problem. First, a sequential solution is defined given one termite. After that, more termites that communicate among themselves are added into the environment featuring a random strategy of exploration. Finally a solution based on stigmergy is proposed. For experiments, a fixed size of the environment is defined ($width = 50$, $height = 50$) and several values of p_f are defined. The idea is to estimate the amount of information that agents can get from the environment.

Sequential Exploration with One Termite

Sequential exploration with one termite is modeled as point of reference given the definition of the world as a matrix. Sequential exploration is a good strategy to explore the world because it implies exploring each location in the world only one time. The termite i knows its current location as $loc_i = (x, y)$ where x corresponds to $rows$ and y corresponds to $columns$. Given this location a simple movement program of the termite is the depicted in Algorithm 3 and Fig 1.

One termite is set to get $data$ in the world of $Props^{50\times 50}$ (size of the world equal to 2500). The termite is located in

```
while (status != Action.DIE) {
        ...
        if (x != width-1) {
                action = right;
        }else{
                action = downright;
        }
        ...
}
```

Algorithm 3: Sequential exploration for termite i

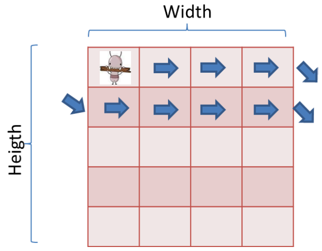


Figure 1: Sequential Exploration Algorithm.

a random location and the world is explored in a sequential way. Given this size of the world, a maximum of 2500 movements is enough for completing information. Sequential exploration experiments were performed 30 times for several values of p_f and results were averaged with the aim of determining how much information a termite can collect in each case.

Table 1 presents the summary of experiments for sequential exploration. Column *Data Collected* displays the average and standard deviation of data collected over the 30 experiments performed. As expected, a greater value in the p_f parameter implies a lesser probability of exploring the entire world. For instance, for a $p_f \geq 10^{-3}$ exploration of the world was never achieved. For the experiments performed with a $p_f = 4 \times 10^{-4}$ a full exploration of the world was achieved only in 12 of the 30 executions. For $p_f = 10^{-5}$ this was achieved in 28 of 30 experiments. The single termite could explore the entire world in a reliable way only if $p_f \leq 10^{-6}$.

p_f	Data Collected	Successful Experiments
10^{-1}	6.8 ± 6.17	$0/30$
10^{-2}	113.27 ± 90.36	$0/30$
10^{-3}	792.2 ± 560.57	$0/30$
4×10^{-4}	1532.93 ± 906.9	$12/30$
10^{-4}	2047 ± 825	$22/30$
10^{-5}	2496.2 ± 14.85	$28/30$
10^{-6}	2500 ± 0	$30/30$
0	2500 ± 0	$30/30$

Table 1: Summary of experiments for crash sequential exploration ($Population = 1, width = 50, height = 50$. A $p_f = 4 \times 10^{-4}$ is added because $1/(width \times height) = 4 \times 10^{-4}$).

Random Exploration with Communication

In these experiments, more than one termite is added to explore the world in a random fashion starting in random locations in the same environment ($Props^{50\times50}$). Termites move in a random way in the environment but have the same p_f. For the same world, experiments were performed with populations of 10, 30 and 50 termites. A maximum of 3000 iterations were defined to get the complete information and each experiment was performed 30 times. Termites can communicate and exchange information with neighbors as specified in the last section *(Termites Definition).*

Levy Walk Exploration With Communication

A Levy walk is a movement process in which a particle makes a sequence of movements in the same random direction during a time length that depends of another uniform random variable. Foraging mechanisms present in some animals appears to obey Levy walks (Benhamou, 2007). Levy Walks have been used for solving the networking coverage problem in robots by moving them until find a location with an acceptable number of neighbors and make connections (Beal, 2013). In this paper, we adapt the motion mechanism of (Beal, 2013) in the following way (Alg. 4): $randomDir()$ returns a random direction $dir \in \{down$, $left$, $right$, up, $upleft$, $upright$, $downright$, $downleft\}$, α is a uniform random number that represents the increment rate of $acumulator$, T defines a threshold of accumulator for generate a new direction dir and Δt defines a increase of α in terms of time. Several experiments using Levy walks were performed using $T = 1$ and $\Delta t = 1$ and varying the other parameters in the same way it was done with random exploration.

```
while $status \neq Action.DIE$ do
    $dir \leftarrow randomDir()$;
    $\alpha \leftarrow U[0, 1]$ //uniform random number;
    repeat
        $move(dir)$;
        $acumulator \leftarrow acumulator + \alpha \cdot \Delta t$;
    until $accumulator \geq T \vee neighbor\_sensor()$;
end
```

Algorithm 4: Reactive Levy walk for termite i (Beal, 2013)

Pheromone-based Exploration

In this approach, each termite determines its movements by using stigmergy inspired by swarms and the Ant Colony

System algorithm (Dorigo and Gambardella, 1997). As a main difference, in this proposal termites are looking for new information (present in other nestmates) instead of looking for food. In this way, termites will have a status determined by the amount of local information that each one has: `SEEKERS` which are termites that look for others for getting new information and to explore locations with more pheromone, and `CARRIERS` that are termites believing they have more information than others and explore world locations with less pheromone. Pheromone value τ_w defined in an environment is used. At the beginning all world locations have a pheromone value of 0.5.

A termite i chooses a direction dir that corresponds to the application of a biased exploration or an exploitation rule depending of a random variable $q \in [0, 1]$ by applying Eq. 1 (Dorigo and Gambardella, 1997):

$$dir = \begin{cases} \text{exploitation rule} & \text{if } q \leq 0.9 \\ \text{biased exploration} & \text{otherwise} \end{cases} \qquad (1)$$

Exploitation rule generates a direction depending of a termite status:

- `SEEKERS`: If a termite is `SEEKER`, this termite will choose the direction with the *maximum* amount of pheromone in its vicinity looking for `CARRIERS`. If all values in the vicinity are the same, a random direction is chosen.

- `CARRIERS`: If a termite is `CARRIER`, this termite will choose the direction with the *minimum* amount of pheromone in its vicinity. If all values in the vicinity are the same, a random direction is chosen.

Biased exploration is a random-proportional rule (Dorigo and Gambardella, 1997) which gives to a termite i a probability of choosing a direction $p_d(x, y)$ depending the amount of pheromone τ_w in its vicinity $neighborhood(i)$ (Eq. 2). $neighborhood(i)$ represent the locations in the Moore neighborhood of i with $r = 1$):

$$p_d(x, y) = \left\{ \frac{\tau_w(x,y)}{\sum_{(k,l)\in \text{neighborhood(i)}} \tau_w(k,l)} \right. \qquad (2)$$

Each time a termite i performs a movement, it updates its local amount of pheromone $\tau_t(i)$ (local update rule of Eq 3) and updates the pheromone in this world location $\tau_w(x, y)$ (global update rule of Eq 4):

$$\tau_t(i) = (\tau_t(i) + 0.01 * (0.5 - \tau_t(i))) \qquad (3)$$

$$\tau_w(x, y) = \tau_w(x, y) + 0.01 * (\tau_t(i) - \tau_w(x, y)) \qquad (4)$$

If a termite i turns into a `SEEKER` or finds new information, its pheromone value is updated to 0 ($\tau_t(i) = 0$). Equation 3 is based on the local update rule of ACS (Dorigo and Gambardella, 1997) and represents a local update rule that makes possible that termite pheromone value increases with the time until a certain point reducing the amount of pheromone of the world (global update rule of equation 4). This influence is reduced with time making pheromone in the world converge to its default value of 0.5. A termite in `SEEKER` state can reach a maximum value of pheromone of 0.5.

If a termite i becomes a `CARRIER` or finds new information, the pheromone of the termite gets a value of 1 ($\tau_t(i) = 1$). Local update rule of Equation 3, produces a decrease of the local amount of pheromone until reaching a minimum value of 0.5. At the same time, a `CARRIER` increases the amount of pheromone in the world (global update rule Eq. 4). This influence is reduced with the time making pheromone in the world converge to its default value (0.5). A termite in a `CARRIER` state can reach a minimum value of pheromone of 0.5.

If one termite has more information than the other termite then it turns into a `CARRIER`; or a `SEEKER` in the other case (Fig 2). In this way and just like in nature, termites are donors to other neighbors in a cascade scheme called *Trophallactic cascade*. Each time that a termite s senses some another neighbor r, s sends its information to r, r merge its information ($I_r = I_r \cup I_s$) and additionally r calculates the difference between information $dif = I_s \setminus I_r$. In this way two following scenarios are possible (Fig 2):

- $dif = \emptyset$ means (in a local way) that r had at least the same information as s, so termite r turns into a `CARRIER` and sets its pheromone value in one $\tau_t(r) = 1$.

- Otherwise (if $dif \neq \emptyset$), r turns into a `SEEKER` and sets its pheromone in zero $\tau_t(r) = 0$.

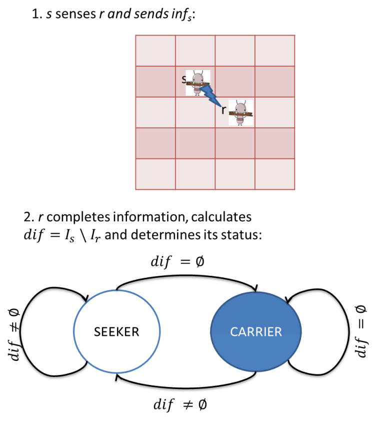

1. s senses r and sends inf_s:
s
r
2. r completes information, calculates dif = I_s \ I_r and determines its status:
dif = ∅
dif ≠ ∅
SEEKER
CARRIER
dif = ∅
dif ≠ ∅

Figure 2: Communication and Status determination of termite r.

Experiments were performed with 10, 30 and 50 termites. All agents at the beginning are SEEKERS, so they start exploring the world and when they make contact with another nestmate information interchange starts. Figure 3 shows how termites start as SEEKERS (white points) explore the world and turn into CARRIERS (blue points) over time. Red locations in the world represent the variation in the amount of pheromone with the time and how agents explore the world. Squares 3, 4, 5 and 6 of Figure 3 depict cases where communication occurs (green circles).

Fig 4 shows how exploration influences the states of the termites between SEEKERS and CARRIERS. The *Termites* axis represents the individuals in the simulation and the *Iteration* axis represents the average round number. After some iterations all the population becomes CARRIERS for all the experiments performed. Bigger populations makes termites turn into CARRIERS in a fast way.

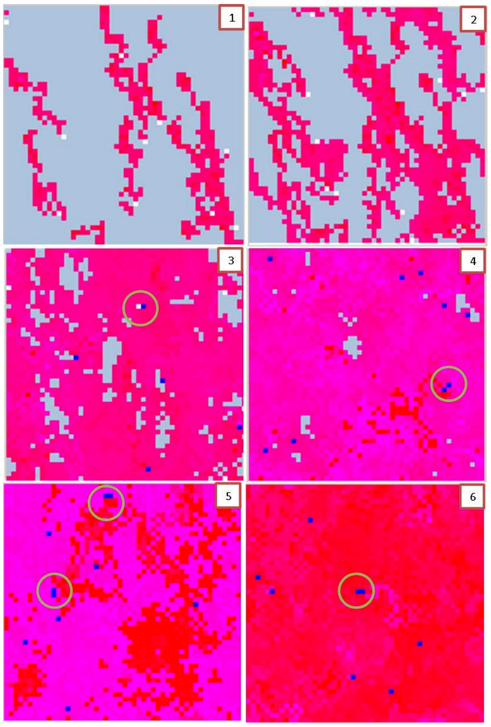


Figure 3: Pheromone exploration with 10 termites, white points are SEEKERS and blue points are CARRIERS.

Results Analysis

Tables 2 and 3 present the results for the experiments of Random Exploration (column Random Expl.). The column *Inf. Col.* shows the average and standard deviation of $data$ collected for agents in the 30 executions. Column *Ag. Compl.* is the average and standard deviation of agents with complete $data$ for the 30 executions of each experiment, Table 3 presents the average round number of the agents that completed information at first place in each experiment.

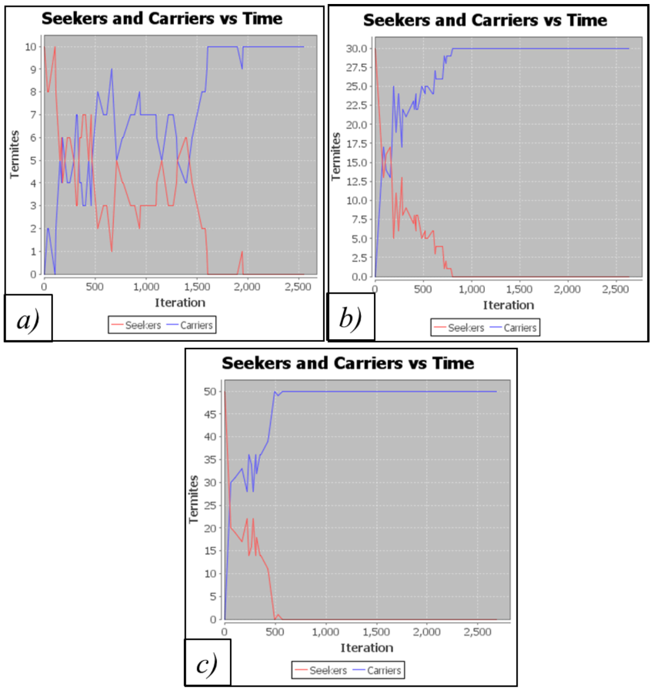


Figure 4: SEEKERS (red line) vs CARRIERS (blue line), $p_f = 0$ $a)$ $Pop = 10$, $b)$ $Pop = 30$, $c)$ $Pop = 50$.

In the experiments performed, a smaller value of p_f means more data obtained and more termites with complete information. However with 10 termites ($Pop = 10$) it was impossible to get the complete $data$ for random exploration. For the given size, random movements and small populations do not warrant exploration of the whole data in the 3000 iterations specified because is difficult for termites to meet and communicate and termites tend to repeat paths. For 30 and 50 termites, for a $p_f \geq 4 \times 10^{-4}$ it is observed more than one agent with all the $data$ information (Agents Complete).

Columns *Levy Walk Expl.* and *Ph. Expl.* of Table 2 presents the results for Levy walks and pheromone exploration respectively. Results show that Levy walks and pheromone exploration work even with 10 termites exploring the world with a $p_f \leq 10^{-4}$ for Levy walks and a $p_f \leq 4 \times 10^{-4}$ in some experiments of pheromone exploration. For 30 and 50 termites the entire information is obtained even with a $p_f = 10^{-3}$. More $data$ is collected as p_f decreases. Bigger populations produce more exploration and a fast information dissemination.

Levy walks are a good technique for exploring new information because the average of information collected is higher compared to pheromone exploration (Table 2). However, the number of agents with complete information before 3000 iterations is bigger for pheromone exploration in 10 and 30 agents and a $pf = 10^{-3}$. In table 3, it is observed

Pop	p_f	Random Expl.		Levy Walk Expl.		Ph. Expl.	
		Inf. Col.	Ag. Compl.	Inf. Col.	Ag. Compl.	Inf. Col	Ag. Compl.
10	10^{-2}	76.68 ± 33.01	0.00 ± 0.00	108.61 ± 52.53	0.00 ± 0.00	122.03 ± 53.08	0.00 ± 0.00
	10^{-3}	897.60 ± 356.59	0.00 ± 0.00	1563.14 ± 335.87	0.00 ± 0.00	1259.91 ± 306.18	0.00 ± 0.00
	4×10^{-4}	1517.67 ± 361.70	0.00 ± 0.00	2073.50 ± 302.45	0.00 ± 0.00	2008.09 ± 314.01	1.57 ± 2.54
	10^{-4}	2268.12 ± 172.21	0.00 ± 0.00	2415.21 ± 114.37	3.70 ± 4.04	2327.36 ± 184.36	7.47 ± 2.50
	10^{-5}	2442.36 ± 75.39	0.00 ± 0.00	2489.92 ± 28.70	6.53 ± 4.46	2489.10 ± 46.04	9.80 ± 0.41
	0	2486.75 ± 6.84	0.00 ± 0.00	2499.66 ± 0.54	7.29 ± 3.98	2500.00 ± 0.00	10.00 ± 0.00
30	10^{-2}	98.69 ± 30.75	0.00 ± 0.00	235.04 ± 79.92	0.00 ± 0.00	174.59 ± 48.76	0.00 ± 0.00
	10^{-3}	1650.54 ± 211.53	0.00 ± 0.00	1991.28 ± 164.40	2.47 ± 4.43	1963.42 ± 190.37	9.03 ± 5.31
	4×10^{-4}	2169.66 ± 132.03	10.07 ± 6.90	2338.12 ± 95.95	20.50 ± 3.15	2218.35 ± 124.45	20.33 ± 2.94
	10^{-4}	2405.94 ± 72.88	25.23 ± 2.53	2451.25 ± 47.00	27.47 ± 1.70	2448.38 ± 51.92	27.83 ± 1.53
	10^{-5}	2491.70 ± 19.91	29.63 ± 0.76	2492.56 ± 22.91	29.67 ± 0.80	2490.70 ± 26.38	29.67 ± 0.61
	0	2500.00 ± 0.00	30.00 ± 0.00	2500.00 ± 0.00	30.00 ± 0.00	2500.00 ± 0.00	30.00 ± 0.00
50	10^{-2}	128.28 ± 30.35	0.00 ± 0.00	428.46 ± 141.93	0.00 ± 0.00	284.50 ± 72.68	0.00 ± 0.00
	10^{-3}	1926.88 ± 156.06	5.90 ± 6.83	2228.73 ± 86.68	22.60 ± 6.42	2051.49 ± 148.74	23.93 ± 4.87
	4×10^{-4}	2230.28 ± 91.13	31.40 ± 4.85	2370.01 ± 57.04	38.07 ± 3.86	2336.00 ± 67.95	40.43 ± 2.80
	10^{-4}	2436.82 ± 41.48	45.17 ± 2.09	2478.72 ± 27.30	47.27 ± 1.87	2459.20 ± 47.88	47.67 ± 1.58
	10^{-5}	2485.10 ± 23.48	49.23 ± 0.90	2496.95 ± 10.79	49.80 ± 0.41	2493.52 ± 16.37	49.63 ± 0.72
	0	2500.00 ± 0.00	50.00 ± 0.00	2500.00 ± 0.00	50.00 ± 0.00	2500.00 ± 0.00	50.00 ± 0.00

Table 2: Summary of average of information collected and number of agents with complete information ($pop = 10, 30, 50$, $width = 50$, $height = 50$, $maxiter = 3000$).

		Average Best Round Number		
Pop	pf	Random Expl.	Levy Walk Expl.	Ph. Expl.
10	10^{-2}	–	–	–
	10^{-3}	–	–	–
	4×10^{-4}	–	–	2162.80 ± 257.22
	10^{-4}	–	2500.81 ± 282.07	1994.39 ± 279.98
	10^{-5}	–	2581.50 ± 219.28	1834.50 ± 224.54
	0	–	2525.78 ± 290.92	1764.35 ± 173.78
30	10^{-2}	–	–	–
	10^{-3}	–	1751.55 ± 516.21	1180.42 ± 332.79
	4×10^{-4}	2038.16 ± 460.05	1074.10 ± 253.87	818.30 ± 90.19
	10^{-4}	1571.40 ± 269.89	925.03 ± 125.36	702.80 ± 54.02
	10^{-5}	1401.70 ± 233.84	871.00 ± 148.31	688.87 ± 48.18
	0	1404.00 ± 231.14	918.57 ± 129.50	691.07 ± 37.52
50	10^{-2}	–	–	–
	10^{-3}	1782.59 ± 570.89	750.70 ± 190.13	626.57 ± 76.69
	4×10^{-4}	1008.00 ± 181.80	584.83 ± 77.98	483.90 ± 35.62
	10^{-4}	874.00 ± 134.37	550.90 ± 133.62	467.87 ± 33.84
	10^{-5}	863.50 ± 144.88	509.03 ± 76.87	458.53 ± 25.03
	0	823.57 ± 96.36	548.77 ± 106.91	451.13 ± 34.85

Table 3: Summary of averages of the number of rounds required for the best agents to collect all information ($pop = 10, 30, 50$, $width = 50$, $height = 50$, $maxiter = 3000$).

that pheromone exploration allows some individuals collect information in a faster way than the other two methods, because the best agents require a small number of rounds in collect all the information.

In the experiments performed, communication is important to reduce the time necessary for getting and disseminating information. Even communication in random exploration tends to reduce the number of rounds necessary for a termite to get all the information from 2500 of sequential exploration with one termite to 823.57 for 50 termites (Table 3). Results with pheromone exploration are even better getting an average number of 451.13 rounds needed for the best termite to collect the complete data without failures ($p_f = 0$).

As expected, bigger populations provide a better performance for exploration and for sharing information in a decentralized, scalable and simple way even with unreliable termites. Additionally, Table 2 shows how Levy walks and pheromone-based exploration imply more resistance to failures compared to random exploration and how a fast data synchronization implies more resistance to failures.

Conclusions and Future Work

In this paper we proposed a solution for obtaining global information via a set of unreliable termites (agents), which explore, get local data and share collected information. The communication mechanism allows neighbors to interchange information, enabling agents to acquire global data as the result of local interactions and cooperation.

It is possible to see how pheromone exploration and Levy walks improve world exploration compared to random exploration. Figure 5 shows how a termite explores the world (a trace is added on visited locations). As expected, pheromone exploration tends to avoid path repetition during exploration vs random exploration where a termite can explore a determined location more than once. Additionally, Levy walks cover an area in a better way than random by maintaining the same direction for some time.

Changes in the status of a termite between `SEEKER` and `CARRIER`, restarts the amount of local pheromone that a termite has, reinforcing the trace and rewarding communication. Future work is intended to test other methods to relate local information with the local update rule of pheromone (e.g. in the way that more local information could represent a stronger trace) and study how a termite's status influences data synchronization and what would happen if perceptions were wrong or if there were changes in the environment and a consensus were required.

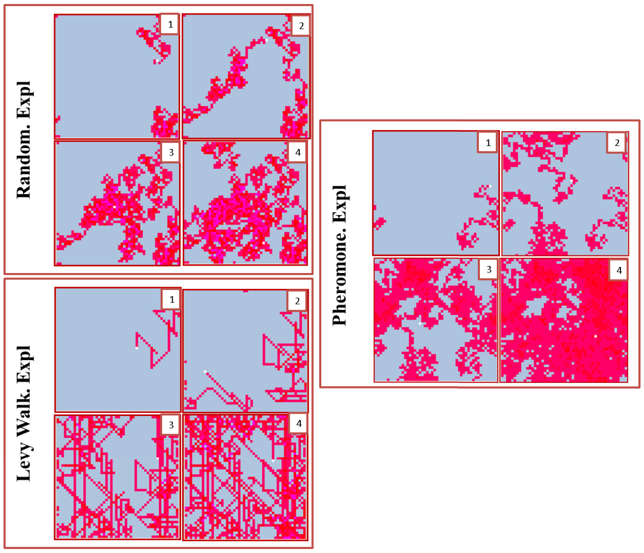


Figure 5: Random, Levy Walk and Pheromone Exploration.

If the same world size is maintained and the population size is increased, a greater amount of information is obtained and a reduction in the average rounds necessary for gathering the whole world information is achieved. Possible next steps include studying the influence of each of these algorithms in terms of number of messages and local information obtained by each termite maintaining the same density of agents and performing experiments with different world sizes.

Acknowledgements

This work was supported by ”Programa de becas para estudiantes sobresalientes de posgrado”, the Alife research group - Universidad Nacional de Colombia, and “Convocatoria del Programa Nacional de Proyectos para el Fortalecimiento de la Investigación, la Creación y la Innovación en Posgrados de La Universidad Nacional De Colombia 2013-2015” code 23418.

References

Aguilera, M. K. (2010). Stumbling over consensus research: Misunderstandings and issues. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5959 LNCS, pages 59–72.

Balaji, P. G. and Srinivasan, D. (2010). An introduction to multi-agent systems. *Studies in Computational Intelligence*, 310:1–27.

Beal, J. (2013). Superdiffusive dispersion and mixing of swarms with reactive levy walks. *International Conference on Self-Adaptive and Self-Organizing Systems, SASO*, pages 141–148.

Beal, J., Correll, N., Urbina, L., and Bachrach, J. (2009). Behavior modes for randomized robotic coverage. *2009 Second International Conference on Robot Communication and Coordination*.

Benhamou, S. (2007). How Many Animals Really Do the Lévy Walk? *Ecology*, 88(8):1962–1969.

Chandra, T. and Toueg, S. (1996). Unreliable failure detectors for reliable distributed systems. *Journal of the ACM (JACM)*, 43(2).

Dorigo, M. and Gambardella, L. M. (1997). Ant Colony System: A cooperative learning approach to the traveling salesman problem. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*.

Doursat, R., Sayama, H., and Michel, O. (2012). Morphogenetic Engineering: Reconciling Self-Organization and Architecture. pages 1–24.

Gray, L. (2002). at Wolfram s New Kind of Science. *Notices of the AMS*, 50:200–211.

Kephart, J. O. and Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1):41–50.

Kshemkalyani, A. and Singhal, M. (2008). *Distributed computing: principles, algorithms, and systems*. Cambridge University Press.

Lalanda, P., Mccann, J. A., and Diaconescu, A. (2013). *Autonomic Computing: Principles, Design and Implementation*. Springer.

Lamport, L. (1998). The part-time parliament. *ACM Transactions on Computer Systems*, 16(May 1998):133–169.

Nedic, A., Ozdaglar, A., and Parrilo, P. a. (2010). Constrained consensus and optimization in multi-agent networks. *IEEE Transactions on Automatic Control*, 55(4):922–938.

Ongaro, D. and Ousterhout, J. (2013). In Search of an Understandable Consensus Algorithm. *Ramcloud.Stanford.Edu*.

Ozdaglar, A. and Nédic, A. (2007). Consensus Problem in Multi-Agent Systems. (July).

Raynal, M. (2013). *Distributed Algorithms for Message-Passing Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rodriguez, A. and Gomez, J. (2011). Programs self-healing over a termites simulator based on language games and evolutionary computing. In Tom Lenaerts, Mario Giacobini, H. B. P. B. M. D. and Doursat, R., editors, *Advances in Artificial Life, ECAL 2011 Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*, pages 664–671, Paris. MIT Press.

Russell, S. and Norvig, P. (2004). *Inteligencia Artificial. Un enfoque moderno. 2da Edición*.

Satzger, B. (2008). *Self-healing Distributed Systems*. PhD thesis, Augsburg University.

Suárez, M. E. and Thorne, B. L. (2000). Rate, Amount, and Distribution Pattern of Alimentary Fluid Transfer via Trophallaxis in Three Species of Termites (Isoptera: Rhinotermitidae, Termopsidae). *Annals of the Entomological Society of America*, 93(Shapiro 1990):145–155.

Tanenbaum, A. and Steen, M. V. (2006). *Distributed systems: principles and paradigms*. Prentice-Hall.

Autonomous lablet locomotion and active docking by sensomotory electroosmotic drive

Abhishek Sharma [1] and John S. McCaskill [1]

[1]Ruhr Universität Bochum, Microsystems Chemistry and BioIT (BioMIP), Universitätsstr. 150, 44801 Bochum, Germany
john.mccaskill@rub.de

Abstract

In this paper, we show how autonomous microscopic CMOS particles, called *lablets,* here at the scale of 100µm, are equipped with an electroosmotic drive, and calculate that this can propel them sufficiently to overcome the strong viscous drag at this length scale, depending on the ionic strength and pH of the solution in which they are immersed. We then use this thrust and the currently fabricated lablet design to show in simulation how programmable movement of lablets can be obtained. The lablets are equipped with chemical sensors and simple programmable functionality that we show can form a substrate for chemotactic behaviour and autonomous docking to a target surface (such as a second lablet). We calculate the concentration field induced by an electrically active lablet and show that other lablets can dock by chemotaxis in this field. Lablets can self-assemble to form compartments, and perform chemical operations, and hence provide a potential electronic-chemical substrate for building artificial cells. The investigation of autonomous motion capabilities is an important basic functionality for particles of this size, where passive Brownian motion is already slow, when motion has been implicated as a fundamental property of early life [1].

Introduction

Electrochemical and electrokinetic locomotion of passive microparticles in solution has recently attracted significant attention, with bimetal rods presenting impressive unswitched locomotion speeds [2], but would need to be combined with extremely miniaturized directional control and intelligent sensing and switching functions to support full robotic potential at this micrometer scale. In a major initiative, our group together with European partners is building smart electrochemical autonomous agents at the scale of 100µm using specially post-processed CMOS chips that we call lablets [3], since they can dock together to form tiny autonomous labs for doing chemistry in a high surface to volume ratio environment.

Previously, we have shown the power of microfluidic-technology incorporating electroosmotic flow [4] (EOF) in addition to electrophoresis in concentrating DNA as well as controlling flows [5,6]. In this paper, we show how autonomous microscopic lablets that we have designed are equipped with an electroosmotic drive, and calculate that this can propel them sufficiently to overcome the strong viscous drag at this length scale, depending on the ionic strength of the solution in which they are immersed. In this simulation, we first calculate the magnitude of the expected electroosmotic thrust for lablets, taking pH effects into account, and then use this thrust in the current lablet design to show how programmable movement of lablets can be obtained. The lablets are equipped with simple programmable functionality, in the first instance this can be described by timed finite state machines, but full microcontroller functionality is achievable if the electronics resolution is enhanced from the current scale of 180nm down to 65nm, using special transistor designs that involve slow switching and low leakage currents being developed by an electronics partner [7].

The lablets are equipped with several sensors that allow pH to be measured in an internal channel at two different points, which also allows gradients to be detected for close proximity docking. However, for chemotactic tracking, gradients are too weak for direct detection using closely spaced sensors, and instead sequential sensor measurements at different positions of a spiraling trajectory show effective chemotaxis of lablets. The paper shows how such motion can be generated with simple two state control of actuation, and explores the extension from lablet chemotaxis to docking, which can allow pairs of lablets to form specific compartments and communicate chemical and electronic information. The paper concludes with a discussion of the implication of lablet docking for living technology processes and a lablet life cycle.

Lablet architecture and design

The basic design of the lablets involves an internal branched channel with the topology of a T and with bent channel ends to permit rotational thrust from electro-osmotically dispelled fluid. A typical lablet is shown in fig. 1. Essentially each lablet consists of a 100x100µm area of a thinned CMOS wafer (40-50 µm), so that two face-to-face lablets can form a cube, with an active enclosed channel network along the mid axis. Alternatively, a single lablet can be fabricated with a laminating film (10-20µm) to enclose the channels. The detailed lablet electronic design and fabrication will be described elsewhere. The lablets are fabricated in standard CMOS (180nm process, Europractice *via* IMEC/TSMC) in collaboration with AIS [7] and T.Maeke at the 20cm wafer scale. Post processing of the lablets includes structured metal recoating of the electrodes (PVD, lift-off, Al to Au), surface topography changes (using the photoresist SU8), specific galvanic post processing of sensor electrodes (IrO2) and supercap inter-digital structures (MnO_2), and lamination. Subsequently, wafers are thinned to 40µm and diced on a supporting substrate to produce the array of lablets. The laminate needs to be photo-structured, also to allow its removal in the saw-lines between lablets to prevent tearing during wafer sawing. The supercap is required to store sufficient charge (from a wireless charging cycle) to support 1000s of electrolyte double layer potential switches. Typical capacitances of 20x20 µm electrodes in solution are 50-100 pF (depending on the electrolyte) whereas the supercap (enclosed on the other side of the lablets) can achieve µF capacitances.

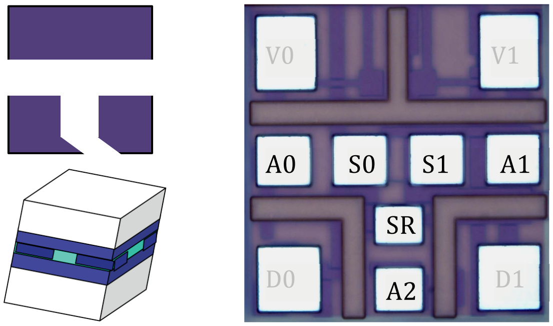


Figure 1. Lablet architecture. Left: Schematic of lablet T channel (bottom) and 3D assembly of a pair of lablets showing enclosed channel exits (light blue-green color). Right: Light microscope image of 100x100µm CMOS lablet substrate, with metal electrodes showing white. Overlaid labels show the three actors A0-2 and the two interior sensors S0-1 with reference electrode SR, in the T shaped channel. The channel is bordered by raised insulating segments (1µm grey) and closed from above either with a laminate or a second docked lablet. The power and docking electrodes (labeled in grey, V0-1,D0-1) are not needed during locomotion.

Lablet electronics is currently equipped with a programmable finite state machine (programmable bit-serially by means of daisy chained flip-flops), and have been made in a combinatorial set of variants, which will be published separately. For the purposes of this article, the following functionality is required:

(i) to be chargeable and run autonomously for a period of 20 min.
(ii) to sense a chemical signal (e.g. pH) and detect if it exceeds a threshold
(iii) to activate either one of two electrode configurations, (a) A0 + A1 (b) A0 + A2, for a fixed duration of time, depending on sensor signal from (ii)
(iv) to be programmed to support a periodically recurring sequence of activity involving simple sequences of actions of type (ii) and (iii).

Electroosmotic thrust calculation

The use of electroosmotic forces to drive fluid through a channel is well known in microfluidics, and was employed by us to fill reaction chambers and filter DNA [5]. The theory of electroosmotic drive is well developed [8,9] so we simply record the main equations used in the standard model for the current work. In this formulation, we consider the ion system to be in equilibrium, so it is primarily governed by Poisson Boltzmann equation. Using these assumptions, we can calculate the electroosmotic thrust produced by coupling with the simplified Navier Stokes equation. In the simplest case where the Debye layer is thin compared with the height of the channel, the electroosmotic velocity is given by

$$v = -\frac{\varepsilon(\zeta_1 + \zeta_2)}{\eta} E_z \tag{1}$$

where the velocity is proportional to the applied field E_z along the channel (in z direction), to the sum of the zeta-potentials of the top and bottom surfaces, and to ε/η *i.e.* the ratio of dielectric constant to dynamic viscosity of the solution. Typical mobility values of 5×10^{-8} m^2/Vs give v =1 mm/s for 2V over a 100µm length channel.

At low ionic strengths, a fuller analysis involves the velocity at any height y in the channel of height h, which can be derived from the force exerted on charges along the channel by means of Poisson's equation

$$F_z = E_z \sigma = E_z \epsilon \frac{\partial^2 \psi}{\partial y^2} = \eta \frac{d^2 v}{d y^2} \tag{2}$$

where ψ is the local electrostatic potential, and σ is the local charge density. The final equality in (2) equates this force to the viscous shear force from the Navier Stokes equation.

Lablet surface channels are predominantly SiO_2 coated and we employ the analysis of [1] [10] for such surfaces[2] [11] in to calculate the zeta potential. The charge density for a partially deprotonated SiO_2/SiOH lablet surface is $\sigma = -e\Gamma_{\mathrm{SiO}^-}$ where $\Gamma = \Gamma_{\mathrm{SiO}^-} + \Gamma_{\mathrm{SiOH}}$ is the total surface density, and

$$\frac{[H^+]_0 \Gamma_{\mathrm{SiO}^-}}{\Gamma_{\mathrm{SiOH}}} = 10^{-\mathrm{p}K_a}, \qquad [H^+]_0 = 10^{-\mathrm{pH}} \tag{3}$$

where the acid concentration near the surface $[H^+]_0$ is related to its value in the bulk, $[H^+]_0 = [H^+]_b \exp\left(-e\psi_0/k_B T\right)$, by the surface potential ψ_0. Furthermore, the potential drop across the Stern layer with capacitance $C = \sigma/(\psi_0 - \psi_d)$ relates this surface potential to the zeta potential $\zeta \equiv \psi_d$ or potential at the inner edge of the diffuse double layer. The six equations in this paragraph can be reduced to a single equation relating σ to ψ_d eliminating $[H^+]_0, [H^+]_b, \Gamma_{\mathrm{SiO}^-}, \Gamma_{\mathrm{SiOH}}, \psi_0$ in favor of known quantities $\Gamma, pH, pKa, C.$ From equilibrium Poisson Boltzmann theory[3], [12], ψ_d is the potential at the inner edge of the diffuse layer and given by

$$\sigma\,(\psi_d) = \frac{2\epsilon\epsilon_0\kappa}{\beta e} \sinh\left(\frac{\beta e \psi_d}{2}\right) \tag{4}$$

and so these two equations for σ, ψ_d can be solved to find $\zeta \equiv \psi_d$. The potential at any height y in the channel of height h can then be found from

$$\psi(y) = \frac{4kT}{e}\left[\left(\tanh^{-1}\left(\tanh\left(\frac{e\zeta}{4kT}\right)\exp\left(-\kappa y\right)\right)\right) + \left(\tanh^{-1}\left(\tanh\left(\frac{e\zeta}{4kT}\right)\exp\left(-\kappa(h-y)\right)\right)\right)\right] \tag{5}$$

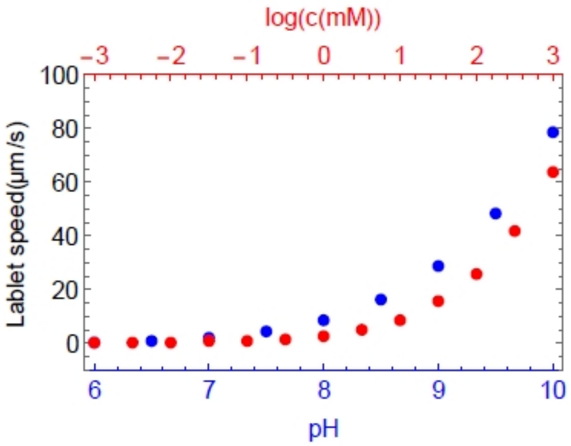


Figure 2. Lablet drift speed under electroosmotic drive. The drift speed for a 100x100µm lablet (hydrodynamic friction set equivalent to a 50µm radius sphere) is calculated as a function of pH (red points) for a concentration of 10mM monovalent salt (e.g. NaCl) and for pH 8 as a function of salt concentration (blue points). Higher drift velocities of several lablet diameters per second can be obtained using a combination of high pH (e.g. 10) and high concentration salt (100mM). Logarithms are base 10. Calculations were performed with the following values of constants, cf. : pH=10, C=2.9 F/m^2, pKa =7.5, Γ= 8/nm^2, bulk ionic concentration 10 mM, applied voltage 2 V, height of channel 1µm.

The solution of eq(3) with slip length b, typically 20nm, is then

$$v(y) = -\frac{\varepsilon\zeta}{\eta}\left[\left(1-\frac{\psi(y)}{\zeta}\right)-\frac{b}{\zeta}\left(-\frac{\sigma}{\varepsilon}\right)\right]E_z \tag{6}$$

and the induced thrust on the lablet can be computed from the mean velocity $\bar{v}$ over the channel as $\rho wh\ \bar{v}^2$

$$F_T = \rho wh\left(\frac{\varepsilon E_z\zeta}{\eta}\right)^2\left(\frac{1}{h}\int\left(1-\frac{\psi(y)}{\zeta}\right)dy-\frac{b}{\zeta}\left(-\frac{\sigma}{\varepsilon}\right)\right)^2 \tag{7}$$

where the solution has density ρ and w is the width of the channel (height h). The linear and rotational drift velocities of lablets (assuming cube = sphere) are

$$v = \frac{\mathrm{F_T}}{6\pi\eta\mathrm{r}} \quad \omega = \frac{\mathrm{F_T}\sin\alpha\,\mathrm{r}}{8\pi\eta\mathrm{r}^3} \tag{8}$$

where the angular velocity in the second equation is equal to the torque on the lablet induced by the thrust from the base of the T in fig. 1 (with fin angle typical value $\alpha = 30°$). We shall use the characteristic values $v = 70\mu\mathrm{m/s}$ and $\omega = 0.6\ \mathrm{rad/s}$ in the remainder of this work, unless otherwise stated. The absolute magnitude of the thrust force is of the order of 0.2 nN, which is less than the 9.8 nN gravitational settling force on a lablet in aqueous solution without buoyancy compensation.

The energy efficiency of the electroosmotic drive can be calculated from the current consumption during operation. There are two contributions to the current: the conductive current and the convective current densities [9]

$$j_{\mathrm{conv}} = 2ec\sinh\left(\frac{e\psi}{k_BT}\right)\frac{eE_z\zeta}{\eta}\left(\left(1-\frac{\psi}{\zeta}\right)-\frac{b}{\zeta}\left(-\frac{\sigma}{\epsilon}\right)\right) \tag{9}$$

$$j_{\mathrm{cond}} = \mu e(n_+ - n_-)E_z = 2\mu e n_0\cosh\left(\frac{e\psi}{k_BT}\right)E_z \tag{10}$$

where both expressions need to be integrated over the channel, c is the bulk ion concentration as in fig. 2 and μ is the salt average ion electrophoretic charge mobility, corresponding to typical room temperature ion diffusion coefficients of $10^{-9}\,m^2/s$, i.e. $3.9\times10^{-8}\,As^2/kg$. The resulting values of the currents at pH 10 and 10mM salt concentration are 39 nA and 5.6 nA. Lablet supercaps can only support powers up to 2nA for 1000s without recharging. An additional factor of 20 is needed for continuous mobile operation at low power densities. Note that the thrust is proportional to the velocity squared while the current consumption is proportional to the velocity, so that this factor can be regained at higher fluid expulsion velocities. Using concentrations of 100mM instead of 10mM at pH 11 increases lablet velocity by a factor of 4 to 5, and current efficiency by over a factor of 2. A factor of roughly 10 in efficiency enhancement is still needed to run locomotion continuously for 20 minutes. This can be achieved using a nozzle structure at the end of channel. For example, reducing the 20µm channel width to a 2 µm opening for 1 µm length will only increase the overall hydrodynamic resistance of the propulsion channel by a factor of 14%, decreasing the interior channel velocity by this amount, while increasing the expulsion velocity and hence the current efficiency by a factor of 8.5. This enables us to reach the locomotion target of navigation for over ¼ hr. Since lablet actuation can be operated for a fraction of the time, resulting in slower motions, efficiency can be regained for our original scenario, and we prefer to continue to work with the slower lablet velocities calculated above, ignoring the nozzle enhancements in the following.

Programmable locomotion of lablets

We first consider the deterministic motion of a lablet under thrust in 2D. Translational and rotational Brownian motion add jitter to this movement and allow a lablet trajectory to explore space more fully. As we saw in Section 2, for 100µm lablets this is indeed a perturbation. Motion in 3D can be achieved through interaction with buoyancy control (e.g. using electro-induced gas formation via electrolysis) see below and discussion/conclusions section.

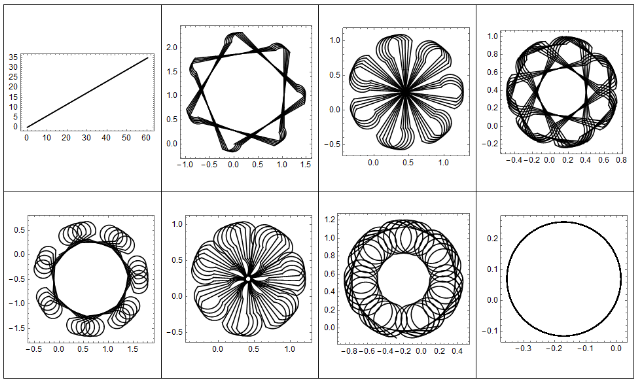


Figure 3. Elementary programmable motions of lablets in 2D. The 8 distinct period-5 two state lablet actuation patterns (see text) are used to generate these trajectories. The duration of the elementary lablet translation and rotation thrusts was same and equal to 3 sec.

Perhaps the simplest form of programmable motion involves switching between just two possible thrust modes, straight and bent, modes 1&2, as described in section 2, involving switching between two possible actuation states. A finite memory of 4 bits for example can then encode a sequence of 4 such binary states, resulting in the 16 elementary motions on cyclic repeat. However not all are different when repeated in a continuous stream, and in fact there are only 6 different trajectories 0000, 0001, 0011, 0101, 0111, and 1111. Similarly, for 5 bits, the 32 elementary motions reduce to eight distinct trajectories 00000, 00001, 00011, 00101, 00111, 01011, 01111, and 11111. We show the trajectories created by these eight standard motions in fig. 3. Each cycle of steps involves a constant net rotation and translation, so that indefinite iteration fills out a circling spiral trajectory provided not all steps are pure translation. Note that depending on the duration of the elementary thrust steps, these deterministic trajectories can approximate space-filling curves. With either longer period sequences or by including a larger number of elementary modes, or by switching between two or more different periodic patterns more complex and divers trajectories can be obtained. Three such different patterns with different resulting curvatures, as shown in fig. 3R are used in the following section.

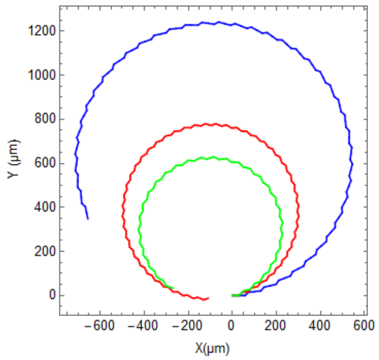


Figure 4. Periodic trajectories using 1,2 and 3 elementary rotations interleaved with translation, which are the basic operations used in the next section for chemotaxis. The steps in the deterministic trajectories are integrated analytically in *Mathematica* both for the pure translation step (mode 1) and the combined translation step (mode 2) and then combined into trajectories.

The equations describing the trajectories of the elementary segments are thus

$$x'[t] = \begin{cases} 2v\cos[\theta[t]] & \text{mode}[t] = 1 \\ v(\cos[\theta[t]](\pm 1 + \sin[\alpha]) - \cos[\alpha]\sin[\theta[t]]) & \text{mode}[t] = 2{,}3 \end{cases}$$

$$y'[t] = \begin{cases} 2v\sin[\theta[t]] & \text{mode}[t] = 1 \\ v(\cos[\alpha]\cos\big[\theta[t]\big] + (\pm 1 + \sin[\alpha])\sin[\theta[t]]) & \text{mode}[t] = 2{,}3 \end{cases}$$

$$\theta'[t] = \begin{cases} 0 & \text{mode}[t] = 1 \\ \omega & \text{mode}[t] = 2 \end{cases} \tag{11}$$

Where mode 1,2 and 3 involve electrodes {A0,A1}, {A0,A2} and {A1,A2} respectively. If the voltages are reversed the signs of all time derivatives also reverses (the lablets move backwards). The magnitude of Brownian fluctuations perturbing these trajectories is shown in the examples of the following section.

The combination of these trajectories with buoyancy control can lead to complex motion in 3D. Although coupling between buoyancy induced motion and lateral motion is possible via hydrodynamics (e.g. disk settling instability) we do not consider such effects in the current work.

Lablet concentration sensing and locomotion

As with mobile single celled organisms, mobility coupled with concentration sensing can allow the organism to navigate to find food in an aqueous environment. Different mechanisms of such chemotaxis are known, including the tumbling mechanism in bacterial chemotaxis [13] and the spiral drift of sperm chemotaxis [14]. Since rotational diffusion is slow on the 50μm radius scale of lablets, compared with driven rotational speeds of *ca.* 0.6 rad/s, the random tumbling of bacteria can be replaced by deterministic directed rotation. Inspired by the elegant chemotactic mechanism for sperm cells [14], which has been modeled as a continuous propagation with curvature *K(t)* governed by

$$K(t) = c_1 + c_2\,\frac{dc}{dt} \tag{12}$$

where c is the chemotractant concentration and c_1 and c_2 are constants, we can attempt to capture part of such a mechanism with a lablet sensor-actuator program.

A very simple program that should be able to perform similar chemotactic lablet locomotion is of the form

```
repeat
1. sensor_value_old = S0-SR        // read sensor value in interior channel
2. activate A0+A1 for $\tau_0$ clock cycles    // activate translation drive
3. sensor_value_new = S0-SR        // read new sensor value
4. activate A0+A2 for $\tau_1$ clock cycles // activate rotation drive
5. if (sensor_value_new>sensor-value_old) then
      activate A0+A2 for $\tau_2$ clock cycles // alternate rotation drive
until end
```

We employed a number of variants of this simple program to investigate lablet chemotactic capabilities for several variants of this algorithm discussed below. We used the steady state solution to the diffusion equation either is spherical or in cylindrical coordinates to define the concentration field. The solution could be expressed analytically with either 1/r or log(r) dependence. In cylindrical coordinates, employed in fig. 5, two boundary conditions were used: concentration c = 100 units at r = 0.1 µm, c = 0.01 at r = 4.2mm. The time for each thrust operation was set to 0.4 secs. The lablets were programmed to sense concentration values after four thrust operations.

Since current lablets are only equipped with a binary threshold (analog-digital conversion) on read, with 2-4 levels, the signals read in steps 1 and 2 are 0-1 or 0-3. In fig. 5 we compare absolute threshold sensing, with very limited bit resolution with continuous sensing, including the effects of sensing noise. Limited sensing resolution restricts chemotactic capabilities when absolute thresholding is employed. A dynamic scaling of sensitivity in analog-digital conversion will be required for robust chemotaxis. If the concentration surrounding lablets are far from the defined threshold, they will continue spanning the region, programmed by the thrust operations.

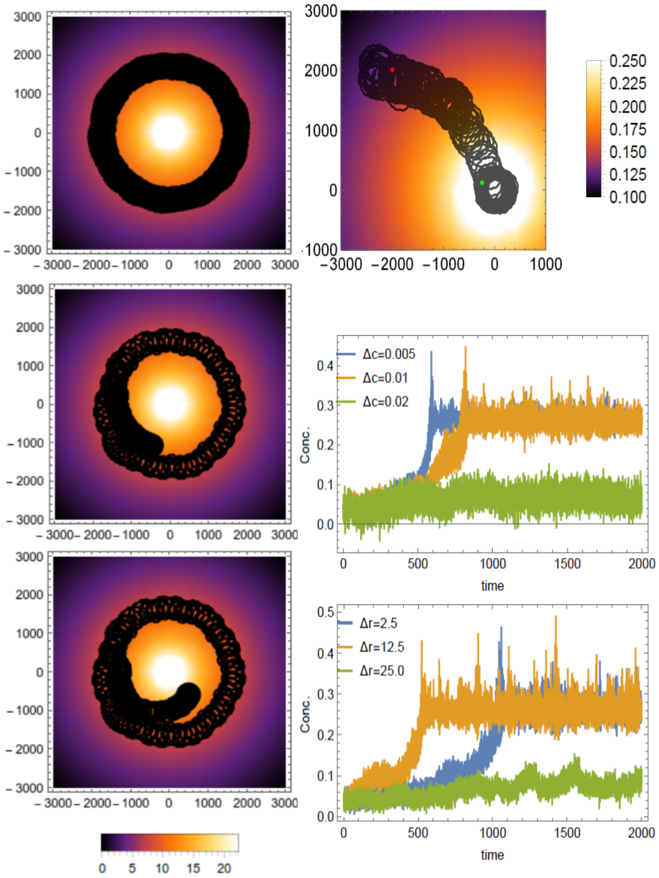


Figure 5. Trajectories of lablets exhibiting chemotaxis. Left: With limited absolute sensor bit resolution: 1,2,4 bits. Top: for 1 bit, {8.5 units}, the lablet rotates around the point source at fixed distance. Mid: for 2 bits, {8.5 units, 10 units}, the lablet slowly comes closer to point source but halts after second threshold condition is met. Bot: for 4 bits, {8.5 units, 10 units, 11.5 units, 13.0 units}, the distance between lablet and point source reduces further till the 4th threshold condition is met. Right: Top: With analog threshold comparison between two measurements. In the presence of chemical, the lablet senses its concentration at two consecutive points {C1, C2} and calculates if (C1 - C2) is +ve or -ve. This defines the next direction of motion (smaller or larger curvature), by changing the number of rotation actuation events. The lablet trajectory is shown from initial position (red point) to final position (green point), sensing chemical concentration. In the absence of chemical gradient, the lablet moves in circle by defined set of default translational and rotational operations. Lower 2 panels: Local concentrations at lablet positions as a function of time, for 3 trajectories with (Mid) increasing sensing noise and (Bot) with increasing distance of lablet from the source.

If the lablets can remember a previous sensed values, instead of comparing concentration with a defined threshold, the algorithm introduced above can be implemented. Based on such comparison, we could define different intervals for rotation operation to locally change the curvature. Using this logic, the lablet could reach the location of highest concentration level. One such case is shown in fig. 5 (right column), using steady state solution from point source. The boundary conditions used were C = 1 units at r = 0.1 µm, C = 0.01 at r = 4.2mm. We studied the performance under different concentration noise levels and noise due to random motion (due to Brownian motion or external perturbations).

Lablet docking

We use external microelectrode array to create dynamically a concentration field (for example pH), so that lablet can sense this field and dock itself on the surface on the microelectrode array. The dynamics of ions created by the electrochemical reactions at the surface of electrode is governed by time dependent Poisson Nernst Planck (PNP) equations. The electrochemistry at the electrode surface is described by Butler-Volmer equation. In the steady state limit, for a binary electrolyte, the governing equations for concentration (c) and potential (ϕ) are given by [15]

$$\nabla^2 c = 0, \qquad \nabla.(c\nabla\phi) = 0 \tag{13}$$

Due to harmonic nature of these equations, they are conformally invariant. For the limiting case of fast reactions (essentially at equilibrium), the boundary conditions also becomes conformally invariant. So, by mapping the analytical solution for one dimensional case, we could create concentration field around two coplanar electrodes. We used Schwarz–Christoffel mapping [16] to map one dimensional solution for concentration and potential between two parallel electrodes to coplanar electrodes. The one dimensional solution for concentration, potential [15] and expression for mapping are given by,

$$c = 1 + J\, Im(f(t)) \tag{14}$$

$$\phi = \log\left(\frac{1 + J\, Im(f(t))}{k(1-J)^2}\right), \qquad \mathrm{J} = \tanh\left(\frac{V}{4}\right) \tag{15}$$

$$\mathrm{f(t)} = \int \frac{K}{(t-t_1)^{1/2}(t-t_2)^{1/2}(t-t_3)^{1/2}(t-t_4)^{1/2}} \tag{16}$$

where, J is the current density scaled to limiting current density, k dimensionless rate constant, V is scaled to thermal voltage and K is a constant determined by matching. So, we used following parameters to create two dimensional profile of concentration field, applied voltage 0.5V, bulk concentration 10mM, characteristic length scale of 10mm, dimensionless rate constant as 0.1. We interpolated the concentration field after mapping the coplanar electrode geometry from the unstructured grid shown in Fig 6A. We defining the initial position of lablet, we sensed the chemical field and perform locomotion similar to the scheme described in the last section. The plotted trajectories in Fig. 6c shows, successful docking of the lablet on the electrode. We observed in the certain limit of chemical noise, lablet, in all the cases, docks on the electrode independent of the starting position of the lablet.

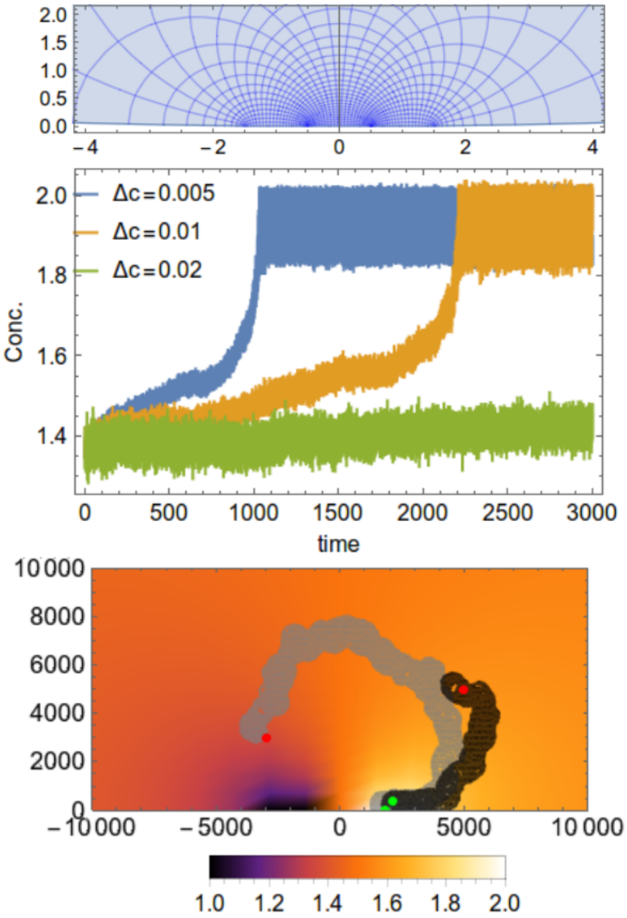


Figure 6. A) Conformal map created by mapping parallel plate electrodes geometry to coplanar electrodes using Schwarz-Christoffel mapping, B) Concentration sensed by lablet over time (starting position shown in C, red point in vicinity of grey curve). The plots show that with increase in the noise in chemical sensing the total time taken to dock increases. C) Trajectory of the lablet from two different starting locations (shown by red points) until it reaches the chemotractant emitting electrode (white, e.g. low pH source).

Work is currently in progress on using various combinations of the forward and reverse modes 1-3 in equation (11) to extend this translational docking of lablets to a lablet emitter chemotractant signal to a full rotational and translational docking model and will be reported at the conference. In further work, we are also exploring the interaction of electroosmotic drive with electrochemical buoyancy control. An initial estimate, indicates that currents of 1nA can create gas bubbles which change the buoyancy of a lablet by 25pN in 10 seconds, which should be sufficient for vertical locomotion with buoyancy compensated lablets. Buoyancy compensated lablets have also been studied experimentally by F. Stepanek using attached light oil droplets (personal communication).

Conclusions: Implications for artificial life

Whereas, much emphasis is placed on the integration of chemical reactions for artificial wet life, and electronic chemical lablets appear an attractive platform to construct working artificial life systems (assuming that the unprogrammed CMOS lablet is regarded simply as a substrate consumable or building block), currently becoming accessible scales of ca. 100μm for smart active chemical particles require special mechanisms of motion to allow lablets to interact. Interaction of lablets with lablet substrate sis essential to a lablet reproduction life cycle, in which a programmed active lablet can dock with a new substrate lablet (unprogrammed) and modify its chemical coating or interior chemicals to allow transmission of program information, prior to undocking release.

Lablet pairing is a novel form of dynamical compartmentation control that can allow lablets to operate on shared chemicals without dilution or interference from the bulk solution. Locomotion is necessary to support pairing, and while this could potentially be achieved by external perturbations (e.g. stirring), it is certainly more efficient to consider autonomous locomoted docking procedures as studied in this paper. In another contribution [17], partners in the MICREAgent project [3], have investigated swarming behavior of lablets assuming an abstract model of sensing and control of motion. The current work complements this study with a more detailed physical investigation of lablet locomotion potential and docking. Whereas much attention has been attracted by the self-propulsion capabilities of oil droplets [1] and bipolar microrod particles [18,19], the current paper shows how programmable control of electrodes can be combined on smart particles at somewhat larger scales to ensure directed locomotion.

In conclusion, lablets with active electronic programs able to interact with chemistry provide a novel substrate for artificial life research. This paper demonstrates that the same mechanisms of electrode actuation and sensing suitable for the control of chemical reactions on lablets can be harnessed to provide autonomously navigating lablets under electroosmotic drive. Careful engineering of channel geometries (e.g. nozzles) will be necessary to ensure sufficient efficiency in thrust for prolonged operation, but the paper establishes that rather simple control mechanisms can then be used to perform complex tasks such as chemotaxis and docking. Docking of lablets allows a novel control of dynamic chemical compartmentation as a substrate for an artificial lablet life cycle as discussed above. Further work in this direction is in progress.

Acknowledgments. Lablet design and fabrication were conducted in collaboration with T. Maeke and P. Wagler (BioMIP) and with the AIS electronics group at RUB: J. Öhm, D. Funke, L. Straczek and P. Mayr, as reported elsewhere. Helpful discussion with U. Tangen and S. Rasmussen is appreciated. This paper was supported by the European Commission under Grant #318671.

References

[1] Hanczyc, M. M. and Ikegami, T. (2010) Chemical basis for minimal cognition. *Artificial Life* 16(3): 233-243.

[2] Moran, J. L. and Posner, J. D. (2011) Electrokinetic locomotion due to reaction-induced charge auto-electrophoresis. *J. Fluid Mech.*, 680: 31–66.

[3] McCaskill, J. S., von Kiedrowski, G., Oehm, J., Mayr, P., Cronin, L., Willner, I., Herrmann, A., Rasmussen, S., Stepanek, F., Packard N.H. and Wills P.R., (2012) Microscale Chemically Reactive Electronic Agents. *Intl. J. Unconv. Comp*. 8: 289-299.

[4] Brask, A., Goranović, G., Jensen, M.J., and Bruus, H. (2005) A novel electro-osmotic pump design for non-conducting liquids: theoretical analysis of flow rate–pressure characteristics and stability. *J. Micromech. Microeng*. 15: 883-891.

[5] Tangen, U. , Wagler, P.F., Chemnitz, S., Goranović, G. Maeke, T., and McCaskill, J.S. (2006) An electronically controlled microfluidics approach towards artificial cells *CompPlexUs* 3: 48 - 57.

[6] Chemnitz, S., Tangen, U., Wagler, P.F., Chemnitz, S., and McCaskill, J.S. (2008) Electronically programmable membranes for improved biomolecule handling in micro-compartments on-chip. *Chemical Engineering Journal*, 135: 276-279.

[7] Oehm, J., Mayr, P., Funke D., and Straczek L., (2015) Work in preparation. AIS, Ruhr Universität Bochum.

[8] Burgreen D. and Nakache, F. R. (1964) Electrokinetic Flow in Ultrafine Capillary Slits*, J Phys. Chem.* 68 (6): 1084–1091.

[9] Diez, F.J., Hernaiz, G., Miranda, J.J., Sureda, M. (2013) On the capabilities of nano electrokinetic thrusters for space propulsion, *Acta Astronautica*, 83**:** 97-107.

[10] Kirby, Brian J. (2013) Micro- and Nano Scale Fluid Mechanics, Transport in microfluidic devices. Cambridge University Press.

[11] Behrens S.H. and Grier, D.G. (2001) The charge on Glass and Silica surfaces. J. Chem. Phys. 115: 6716

[12] Butt, Hans-Jurgen, Graf, Karlheinz; Kappl, Michael (2006). Physics and chemistry of interfaces. Germany, Wiley-VCH. pp. 45, 55, 56, 76–82. ISBN 978-3-527-40629-6.

[13] Larsen, S.H., Reader, R.W., Kort, E. N. , Tso, W.-W. and Adler J. (1974). Change in direction of flagellar rotation is the basis of the chemotactic response in Escherichia coli. *Nature* 249: 74–77.

[14] Alvarez, L., Dai, L., Friedrich, B.M., Kashikar, N.D., Gregor, I., Pascal, R. and Kaupp, U. B., (2012) The rate of change in Ca2+ concentration controls sperm chemotaxis. *J. Cell Biol.*, 196(5): 653–663.

[15] Bazant, M.Z. (2004) Conformal mapping of some non-harmonic functions in transport theory. *Proc. R. Soc. Lond. A* 460: 1433–1452.

[16] Schinzinger R. and Laura, P. (1991) Conformal mapping: methods and applications, p. 60, Elsevier, Amsterdam.

[17] Tangen, U. and Rasmussen, S.R. (2015), in current volume.

[18] Paxton, W.F. , Kistler, K.C. , Olmeda, C.C. , Sen, A., St Angelo, S.K. , Cao, Y., Mallouk, T.E. , Lammert, P.E. and Crespi, V.H. (2004) Catalytic Nanomotors: Autonomous Movement of Striped Nanorods. *JACS* 126(41): 13424–13431.

[19] Soler, L. and Sanchez, S. (2014) Catalytic nanomotors for environmental monitoring and water remediation. *Nanoscale* 6(13): 7175.

A Predator-Prey Scenario in a Virtual Ecosystem

Nesrine Ouannes[1], NourEddine Djedi[1], Yves Duthen[2] and Hervé Luga[2]

[1]LESIA Laboratory/ Biskra University, BP 145 RP Biskra, Algeria
[2]IRIT Laboratory/ Toulouse 1 University- CNRS - UMR 5505 Toulouse, France
nesrineouannes@gmail.com

Abstract

One of the main topics in artificial life is the design of systems that exhibit some characteristics of living organisms. Among the great variety of biological systems that inspire and guide these researches and according to Bedau et al. (2003), three broad areas can be identified depending on their basic elements: (a) At the microscopic scale, chemical, cellular and tissular systems; Wet ALife synthesizes living systems out of biochemical substances, (b) At the mesoscopic scale, organismal and architecture systems; or the Soft ALife that uses simulations or other purely digital constructions that exhibit lifelike behavior, (c) At the macroscopic scale, collective and societal systems. In our model we try to blend at least (b) and (c) in the same simulation. In this abstract, we propose architecture to simulate a virtual ecosystem and present extended results. This ecosystem is populated with 3D artificial creatures that have to use a simple predator-prey scenario.

Artificial behaviors are developed in order to control artificial creatures. The artificial creatures living in the ecosystem are divided into four classes: producers (plants), 2 kinds of consumers (herbivores and carnivores) and decomposers such as bacteria (Ouannes et al, 2014). First, we studied the behavior of herbivorous creatures (Ouannes et al, 2012), which feed on available resources in their environment.

In this part of our ecosystem, we present a controller model, which controls physically the simulated creatures in a biologically inspired manner; by evolving the neural connection weights of a neural network to obtain some emerged strategies of a predator prey, in addition to foraging behaviors (Ouannes et al, 2012). Here, the resulting behaviors are obtained from a predator prey encounter in a shared environment, the physical parameters and those of evolution are the same used in our previous work (Ouannes et al, 2012), but with two populations coevolved using external fitness functions that have opposite goals. The resulting controllers have been evolved in accordance to the physical laws and generate motion, taking into account the Newtonian dynamics (see figure 1).

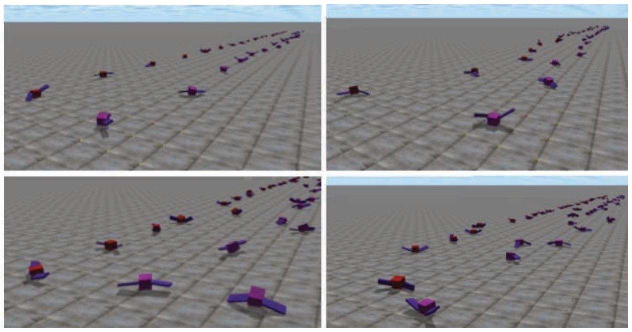


Figure 1: Four screenshots of the simulation on different time intervals.

Two populations of 100 controller genomes of each species (predator or prey) are co-evolved through 5000 generations. To evaluate the fitness of a predator (prey), we test a predator prey pair in the physics-based simulator for a constant simulation time using the RNN weights coded by each chromosome.

$$\boldsymbol{F_{Pred}= \alpha 1 * (R_0 - R_f)/R_0 \qquad if\ R_f < R_0 \qquad 0 \qquad else \qquad (1)}$$

Where R_0 is the initial distance between the predator and the prey, R_f is the final distance. $\alpha1$ (500) presents the normalization factor, which depends on the prey's fitness value.

$$\boldsymbol{F_{Prey}= \alpha 2 * (L/L_n) \qquad if\ \ L_n < L \qquad 0 \qquad else \qquad (2)}$$

The fitness of the prey is defined in the equation (2), where the value of this function is to be maximized. (L) is the distance traveled by running away from the enemy, (L_n) is the initial distance. $\alpha2$ (50) is the normalization weight.

The values obtained from this function will be used to compare the individual's performance between different generations and to generate statistical data. The co-evolutionary process used in this experiment was able to successfully produce stable walking movements that allowed the predator (prey) to move towards (away from) its enemy. Our best-evolved creatures of the two species were able to emerge some defense strategies as presented in the *fitness graph* (see figure 2). Such populations can be used to study more sophisticated behaviors in a virtual ecosystem; where the best creatures of both species will be placed into an environment with energy. The species dynamics can be analyzed in an ecological context.

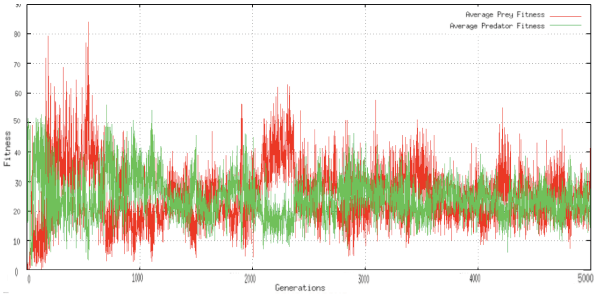


Figure 2: The graph of fitness values of the predators and the preys.

References

M. A. Bedau, "Artificial life: Organization, adaptation and complexity from the bottom up", Vol. 7, n° 11, pp. 505–512, 2003.

N. Ouannes, N. Djedi, Y. Duthen, and H. Luga (2014). Modeling a bacterial ecosystem through chemotaxis simulation of a single cell. AROB Journal Springer Japan, 19(4) : 382-387.

N. Ouannes, N. Djedi, Y. Duthen, and H. Luga (2012). Following food sources by artificial creature in a virtual ecosystem. VIRTUALWORLDS – Artificial Ecosystems and Digital Art Exploration, pages 99-116.

Cooperative Object Transport Using Evolutionary Swarm Robotics Methods

Muhanad H. Mohammed Alkilabi[1,2], Chuan Lu[1], and Elio Tuci[1]

[1]Computer Science Department, Aberystwyth University, Aberystwyth, SY23 3DB, UK
[2]Computer Science Department, Kerbala University, Kerbala, Iraq
mhm1@aber.ac.uk, cul@aber.ac.uk, elt7@aber.ac.uk

Abstract

This paper describes a set of simulations in which autonomous robots are required to coordinate their actions in order to transport a cuboid object that is too heavy to be moved by single robot. Robots' controllers are synthesised using artificial evolution and dynamic neural networks. We compare two different types of robots: in the *NT-condition*, the robots are equipped with a camera and proximity sensors. In the *T-condition*, the robots have additional torque sensors. The result shows that best evolved groups of the *T-condition* outperform those of the *NT-condition*. Moreover, we show that the best evolved groups can adapt to variability in size and weight of the object as well as to the small variability in the cardinality of the group. We also show that simple forms of recruitment behaviour emerges without being selected for during evolution. This work unveils interesting relationships between design choices and characteristics of the evolved solutions, and it contributes to develop design guidelines for engineering robust and successful swarm robotic systems.

Introduction

In multi-robot systems, group transport refers to the coordinated action of a group of robots in order to collect and retrieve objects that are too heavy to be transported by a single robot. In swarm robotics, group transport is a particularly challenging task because the coordinated response of the group should be achieved without centralised control, global information, and non-stigmergic forms of communication (see Dorigo and Şahin, 2004). Under these conditions, natural swarms, and in particular ants manage to transport large food items, by overcoming problems such as the distributions of porters around the burden, and the coordination of the forces in order to avoid to work against each other (Sudd, 1965).

Swarm roboticists aim to mimic the behaviour of natural swarm by looking for the individual rules that generate such robust group-level response. For example, in (Berman et al., 2011), the authors observed a particular species of ants (*Aphaenogaster cockerelli*) in order to extract and reproduce in a simulated swarm robotic system those rules that govern the individual action during group transport. Other studies have looked at different aspects that can bear upon the efficiency of the group during transport. In (Dorigo and et al., 2013), the authors investigated the effects of the degree of relatedness of the individuals by proposing a swarm robotic system in which the group transport is accomplished by multiple groups of robots, in which each group is homogeneous (i.e., all the robots are the same), but robots of different groups have different structural and functional characteristics. In (Wang et al., 2004), the authors look at the effects produced by the introduction of hierarchies in the group (e.g., leader and followers). In (Kube and Bonabeau, 2000), the authors extend a work originally developed in (Kube and Zhang, 1993) to look at the efficiency of group transport strategies by varying the shape of the object as well as the cardinality of the group (i.e., the number of robots in the group). The results of this study indicate that group transport strategy becomes progressively less efficient when the ratio between number of robots and length of surface on which to exert forces increases.

Sensitivity to size and shape of the object, in addition to undesired negative effects during test on scalability with respect to group cardinality seem to be the major obstacles to the design of efficient group transport strategies in swarm robotics systems (see Groß and Dorigo, 2009). To overcome these limitations, roboticists have been exploring alternative solutions. For example, in (Mataric et al., 1995) the authors look at the effects of direct communication. Their study shows that a two-robot group capable of coordinating the action through a dedicated communication protocol outperforms a two-robot group that can not use direct communication. The aim of communication was to help the robots to complement each other partial sensory view of the object. Communication in group transport scenarios seems to be an effective tool to boost the group performances. However, the use of dedicated communication protocol bears upon the robustness and scalability of the system (see Brambilla et al., 2013). With the aim to improve the efficiency of group transport, a series of studies investigated the use of more sophisticated forms of transport, such as assembly that requires the robots to physically grasp either the object or other group mates which are either directly or indirectly connected to the object. The authors in (Groß and Dorigo, 2004, 2008) showed that group transport strategies based on assembly are robust and scalable with respect to various characteristics of the object as well as to the cardinality of the group.

The work on assembly above mentioned is one of the few among those dedicated to group transport, in which the authors opted for Evolutionary Swarm Robotic (ESR) methods. ESR is a design method based on the use of artificial evolution to find set of parameters for artificial neural networks that guide the robots to the accomplishment of their task. ESR can be used to synthesise individual mechanisms underpinning complex group responses such as those required in group transport. Our long term objective is to show that ESR can help to overcome some of those limitations currently observed in swarm robotic systems engaged in group transport scenarios, through the design of self-organised and adaptive transport strategies. Recent studies have emphasised the need to engineer self-organisation through investigations focused on the effects of design choices offered by the evolutionary approach on the quality of the solutions (Doncieux and Mouret, 2014; Trianni, 2014). This study aims to contribute to the development of a principled methodological approach to the design of group transport strategies using ESR. In particular, we investigate the effects produced by the characteristics of the robot sensory apparatus on the design of group transport strategies in simple swarm robotic systems that can only push objects.

Our experimental design is made of two conditions: in the *T-condition* the simulated robots are equipped with a camera for colour perception, proximity sensors, and torque sensors; in the *NT-condition* the robots can only use the camera and proximity sensors. The results of the study show that artificial evolution exploits the additional torque sensors by developing more robust and more effective group transport strategies which are robust to variations in size and mass of the object as well as with respect to the cardinality of the group. Moreover, the perceptual apparatus of those robots equipped with torque sensors generate adaptive responses that have not been selected during evolution. In particular, the robots are capable of using behavioural responses, originally evolved to avoid deadlocks during coordination, to develop recruitment behaviour.

We are aware that one of the criticism moved to the evolutionary approach concerns the loss of performance during transfer of the evolved solutions on real hardware (see Francesca and et al., 2014). Although our work is in simulation, our robot model and the physics of the robot-world interactions have been carefully developed in order to facilitate the transfer to real hardware. We are currently testing and comparing performances of the system in simulation and on real robots. At the time of writing, this comparative analysis has not been completed yet. However, as shown in `http://users.aber.ac.uk/elt7/ECAL2015_suppMat/`, the results of initial tests demonstrate that the best evolved solutions can be successfully ported on real robots. Thus, we believe that ESR approach can be an effective design method to provide swarm robotic systems the mechanisms required to mimic the group transport behaviour observed in natural swarms. This work unveils interesting relationships between design choices and characteristics of the evolved solutions, and it contributes to develop design guidelines for engineering robust and successful swarm robotic systems.

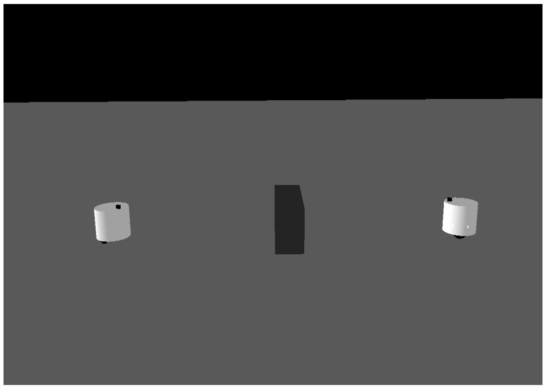


Figure 1: Snapshot taken from the simulator showing the robots and the object.

The Task and the Simulation Model

This study focuses on an object-transport task in which a group of two robots is required to push an elongate cuboid object which is too heavy to be moved by a single robot. The robots are initially positioned in a boundless flat arena at 50cm from the object (see Fig. 1). From their initial positions, the robots can perceive the object with their camera, and when sufficiently close to it, they can sense it with their infra-red sensors. The task requires the robots to independently search for the object and move towards it. Once in the proximity of the object, the robots have to coordinate their actions in order to push the object by exerting the forces required to transport it as far as possible from its initial position.

To take into account the dynamic aspects of this group transport scenario (e.g., forces, torque, friction, etc.), the agents and their environment have been simulated using *Bullet* physics engine. The object has a cuboid shape (30cm length, 6cm width, 6cm height) with a mass of 280g. Our simulation models an e-puck robot (see Mondada and et al., 2009). The robot model consists of three rigid bodies, a cylindrical chassis (3.55cm radius, 6.2cm height 200g mass), and two motorised cylindrical wheels (2.05cm radius, 0.2cm height, 20g mass) connected to the chassis with hinge joints. Both wheels can rotate forwards and backwards at a

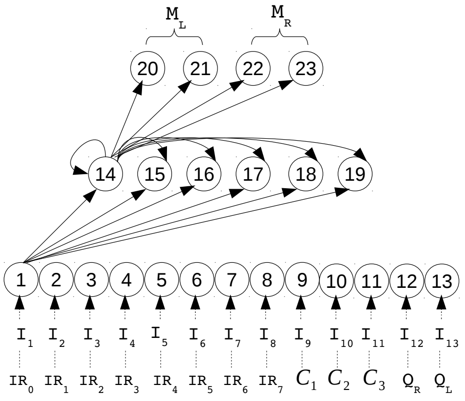


Figure 2: The robot's controller for the *T-condition*. Continuous line arrows indicate the efferent connections for only one neuron of each layer. Hidden neurons receive an afferent connection from each input neuron and from each hidden neuron, including a self-connection. Output neuron receive an afferent connection from each hidden neuron. Sensors to sensor neurons correspondence is indicated underneath the input layer.

maximum speed of 8cm/s.

Each robot is provided with eight infra-red sensors (IR^i with $i = \{0, .., 7\}$), which give the robot a noisy and non-linear indication of the proximity of an obstacle (e.g., the object or another robot). The IR sensor values are computed using a non-linear regression model of the sensor readings collected from the real e-puck (see Michel, 2004, for details). Each robot is also equipped with a camera that can perceive coloured items (i.e., the object which is green, or robots which are all red). The camera has a receptive field of 30°, divided in three equal sectors C_i, with $i = \{1, 2, 3\}$, each of which can return one of four possible values: 0 if no item falls within the sector's field of view; 0.4 if one or more red items are perceived; 0.7 if a green item is perceived; 1.0 if red and green items are perceived. The camera can detect coloured objects up to a distance of 50cm.

This study is made of two experimental conditions: the *T-condition* and the *NT-condition*. In *T-condition*, the robots are equipped with additional torque sensors, placed on the robot left and right wheel (Q_R and Q_L). In *NT-condition* the robots do not have torque sensors. A high level of random noise applies to all sensors and motors to guarantee that the controllers transfer to the real robot with no loss of performance (see also Jakobi et al., 1995).

The Controller and the Evolutionary Algorithm

The robot controller is composed of a continuous time recurrent neural network (CTRNN) of $N = 13$ sensor neurons for controllers of the *T-condition*, and $N = 11$ for controllers of the *NT-condition*. All controllers have 6 internal neurons, and 4 motor neurons (see Beer and Gallagher, 1992, and also Fig. 2, which illustrates the structure of the network). The states of the motor neurons are used to control the speed of the left and right wheels as explained later. The values of sensory, internal, and motor neurons are updated using equations 1, 2, and 3.

$$y_i = gI_i;\ i \in \{1, ., N\}; \tag{1}$$

$$\tau_i \dot{y}_i = -y_i + \sum_{j=1}^{j=N+6} \omega_{ji}\sigma(y_i + \beta_j);\ i \in \{N+1, ., N+6\}; \tag{2}$$

$$y_i = \sum_{j=N+1}^{j=N+6} \omega_{ji}\sigma(y_j + \beta_j);\ i \in \{N+7, ., N+10\}; \tag{3}$$

with $\sigma(x) = (1 + e^{-x})^{-1}$. In these equations, using terms derived from an analogy with real neurons, y_i represents the cell potential, τ_i the decay constant, g is a gain factor, I_i with $i = 1, .., N$ is the activation of the i^{th} sensor neuron (see Fig. 2 for the correspondence between robots sensors and sensor neurons), ω_{ij} the strength of the synaptic connection from neuron j to neuron i, β_j the bias term, $\sigma(y_j + \beta_j)$ the firing rate f_i. All sensory neurons share the same bias (β_I), and the same holds for all motor neurons (β_O). τ_i and β_i of the internal neurons, β_I, β_O , all the network connection weights ω_{ij}, and g are genetically specified networks' parameters. At each time step, the output of the left motor is $M_L = f_{N+7} - f_{N+8}$, and the right motor is $M_R = f_{N+9} - f_{N+10}$, with M_L , $M_R \in [-1, 1]$. Cell potentials are set to 0 when the network is initialised or reset, and equation 2 is integrated using the forward Euler method with an integration time step T = 0.1.

A simple evolutionary algorithm using roulette wheel selection is employed to set the parameters of the networks (Goldberg, 1989). The population contains 100 genotypes. Generations following the first one are produced by a combination of selection with elitism, recombination, and mutation. For each new generation, the five highest scoring individuals (the elite) from the previous generation are retained unchanged. The remainder of the new population is generated by fitness proportional selection from the 60 best individuals of the old population. Each genotype is a vector comprising real values coding for the network's connection weights, decay constants, bias terms and gain factor. Initially, a random population of vectors is generated by initialising each component of each genotype to values chosen uniformly random from the range $[0, 1]$. New genotypes, except the elite, are produced by applying recombination and mutation. Each new genotype has a 0.3 probability of being created by combining the genetic material of two parents. During recombination, one crossover point is selected. Genes from the beginning of the genotype to the crossover point is copied from one parent, the other genes are copied from the second parent. Mutation entails that a random Gaussian offset is applied to each real-valued vector component encoded in the genotype, with a probability of 0.04. The mean of the Gaussian is 0, and its standard deviation is 0.1. During evolution, all vector component values are constrained to remain within the range $[0, 1]$.

The Fitness Function

During evolution each group undergoes a set of $E = 8$ evaluations or trials. A trial lasts 60s (i.e., 600 simulation steps with 1 stimulation step corresponding to 0.1s). At the beginning of each trial the controllers are reset, and the robots are positioned in the arena. Each trial differs from the others in the initialisation of the random number generator, which influences all the randomly defined features of the environment, the noise added to sensors, and the robots initial position and orientation. The robots initial relative position with respect to the object is an important aspect which bears upon the complexity of this task. This is because the robots initial position contributes to determine the orientation with which the robots approach the object and consequently the nature of the manoeuvres required by the agents to coordinate and synchronise their actions. During evolution, the robots starting positions correspond to randomly chosen points on a circle's circumference of 50cm radius that has the object in it's centre. First, one point is randomly chosen and one robot is positioned in the arena. For the second robot, we proceed in the following: in half of the trials, the second robot is positioned at 180° with respect to the first one; in the other half of the trials, the second robot is randomly positioned either on the right or on the left of the first one, at an angular distance randomly chosen in [30°, 40°]. Each robot is randomly oriented in a way that the object can be within an angular distance of ±60° from its facing direction. These criteria should generate the required variability to develop solutions that are not sensitive to the robots initial positions.

In each trial (e), an evaluation function F_e rewards groups in which the robots remain closer to the object, and transport the object as far as possible from its initial position. F_e is computed in the following:

$$F_e = \sum_{t=0}^{T} \left(f_t^1 + f_t^2\right) + f^3; \text{ with } T = 600; \quad (4)$$

$$f_t^1 = \sum_{r=1}^{R} IR_1^r + IR_2^r + (1 - d_r); \text{ with } R = 2; \quad (5)$$

$$f_t^2 = O^{velocity}; \quad (6)$$

$$f^3 = \Delta O^{position}; \quad (7)$$

t is the current time-step; d_r is the Euclidean distance between the centroid of robot r and the centroid of the object. d_r is set to zero if the robot gets closer than 20cm to the object. IR_1^r and IR_2^r are the readings of the front infra-red sensors of robot r. $O^{velocity}$ is the linear velocity of the object. $\Delta O^{position}$ is the Euclidean distance between the position of the object's centroid at the beginning and the end of the trial. f^1 rewards groups in which the robots approach and collide with the object. f^2 rewards groups that transport the object at maximum speed regardless of the object's trajectory. f^3 rewards groups that transport the object. The aim of f^2 is to favour those transport strategies that, being $\Delta O^{position}$ equal, generate longer trajectories. The fitness of a genotype ($\bar{F}$) is the average team evaluation score after it has been assessed $E = 8$ times: $\bar{F} = \frac{1}{E} \sum_{e=1}^{E} F_e$.

Results

For each of the two experimental conditions, 26 differently seeded evolutionary simulations have been run for 4000 generations. At the end of the evolutionary phase, for each experimental condition, we have re-evaluated the best groups of each generation of the last 3000 generations of each run. This post-evaluation test aims to evaluate the effectiveness of the group transport strategies in a set of operating conditions larger than the one experienced during evolution, in which the object length, the object mass, the group cardinality (i.e., the number of robots in a group), and the robots initial relative positions are varied. During post-evaluation, homogeneous groups of 2, 3, and 4 robots are required to transport objects of length 20cm, 30cm, and 40cm, and of 3 different masses, each of them sufficiently heavy to require a combined effort of all the agents of the group to transport the object. For each operating condition (i.e., for each combination of object mass, object length, group cardinality), the evaluation is repeated 21 times (trials), by varying the robots initial relative positions according to the 7 different patterns shown in Fig. 3. Each pattern is repeated 3 times changing the robots initial orientation. Each post-evaluation trial lasts 60s (i.e., 600 simulation cycles).

Fig. 4 shows the performances of the best group of each

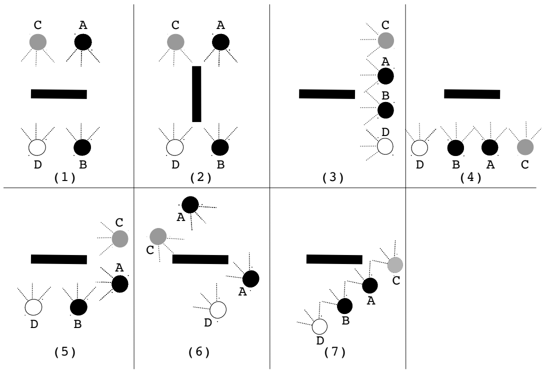


Figure 3: Starting positions during post-evaluation test. Black circles indicate starting positions of two-robot groups. For thee-robot and four-robot groups, the starting positions can be obtained by including the grey and the white circles, respectively. Thick lines represent the object.

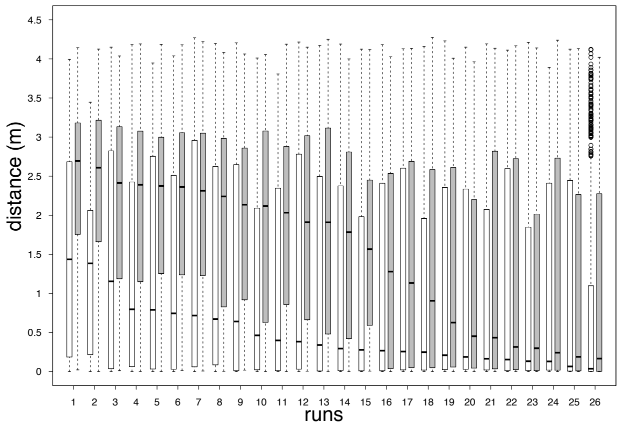


Figure 4: Graph showing the distance (in meters) the objects have been transported by each of the best evolved groups of each evolutionary run of each experimental condition, during 567 trials (7 different robots initial positions, 3 object lengths, 3 object masses, 3 group cardinality, and 3 initial robot orientation). White boxes refer to groups of condition *NT-condition*; grey boxes refer to groups of condition *T-condition*. Each point in the box refers to the group performance in a single trial. Boxes represent the inter-quartile range of the data, while dashed horizontal bars inside the boxes mark the median value. The whiskers extend to the most extreme data points within 1.5 times the inter-quartile range from the box. Empty circles mark the outliers.

run in each experimental condition. The runs are ranked in descending order of performance, with group of run n. 1 being the one with the highest median. As expected the groups that can use torque sensors (see Fig. 4, grey boxes) have a better performance than the groups that do not use torque sensors (see Fig. 4, white boxes). By visually inspecting the groups' behaviour we noticed that this performance difference can be explained with reference to how the groups of the *T-condition* use the additional sensory information provided by the torque sensors.

The robots without torque sensors move towards the object and keep on pushing it at maximum speed. These robots do not have any means to distinguish between those circumstances in which the forces exerted on the object move it from those that don't. Thus, they "blindly" keep on pushing the object even when the forces do not produce any significant object movement. This may generate deadlocks (e.g., robots working against each other) from which the robots may not be able to recover. Alternatively, the combination of the direction of pushing and minimal rotational movements of the object help the robots to recover and eventually to coordinate their effort to transport the object. The robots are more likely to incur in deadlock situations when they start from the opposite sides of the object (e.g., starting position n. 1 in Fig. 3). However, in these more difficult starting conditions, robots of the most successful groups, can avoid deadlocks by getting closer to each other before reaching the object. This happens occasionally when they see each other while they are approaching the object.

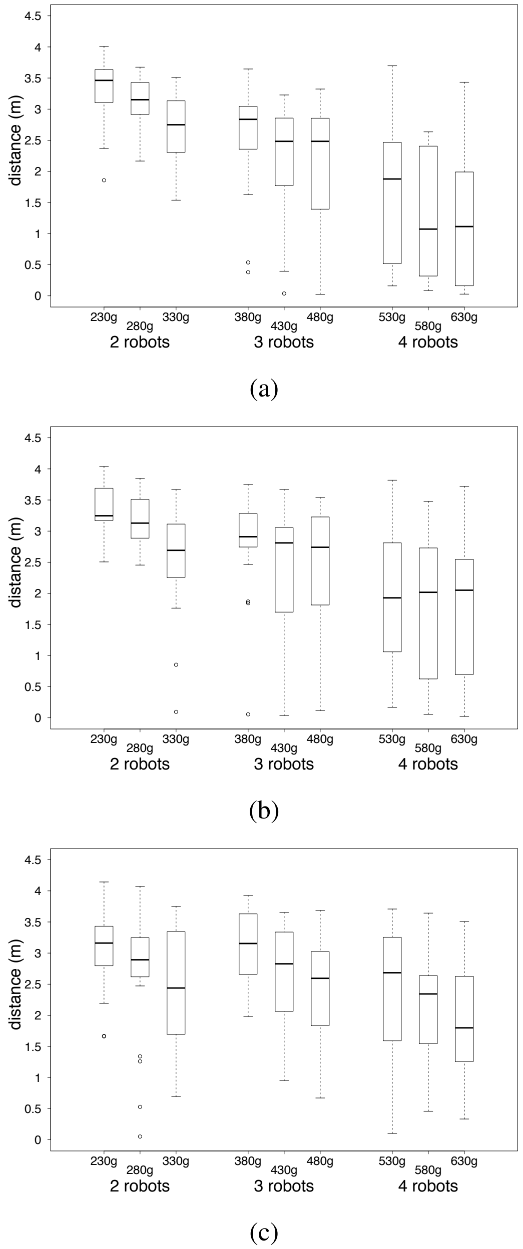
The robots with torque sensors can use the additional sensory information as a valuable feedback to perceive when the forces exerted on the object do not produce any movement. If torque sensors indicate that the robot's action do not produce any benefit, then the robot stops pushing the object, moves first few centimeters away from it, and then towards the object again but in a slightly different position. This manoeuvre is repeated until either both robots place themselves in a position relative to the object from which they can effectively transport the object, or until one robot sees one of its group mates. In this case, the robot moves closer to the group mate/s. When all robots get closer to each other on the same side of the object they initiate the transport. The group transport can be achieved either with all robots pushing the object side by side, or with one or two robots pushing the object and the others pushing those in contact with the object. In summary, the torque sensors help robots to avoid deadlocks and improve the group performance.

On the Robustness of the Best Groups of the T-condition

The results illustrated in previous Section indicate that artificial evolution can generate successful homogeneous groups of robots by designing neural mechanisms that exploit the potentialities of the robot sensory apparatus to develop effective group transport strategies. In this Section, we show further data which illustrate how effective the best evolved group of the *T-condition* is in dealing with conditions never encountered during evolution.

Graphs in Fig. 5 summarise the results of the best evolved controllers of *T-condition* during the previously illustrated post-evaluation test. Graphs in Fig. 5a, 5b, and 5c refer to results with objects of 20cm, 30cm, and 40cm length, respectively. Recall that the evolutionary conditions concerned two-robot groups required to transport a 30cm object of 280g. Thus, in Fig. 5b the second box represents the performance of the system when re-evaluated in evolutionary conditions. This box can be used as a term of comparison to estimate the robustness of the robotic system in all the other test conditions.

For each group cardinality, and for each object length, there is a common trend which indicates that heavier the objects, shorter the distances the object has been transported. The other important aspect that emerges from these graphs is the drop in performance when four-robot groups are required to push 20cm length objects. The object dimension compared to the group cardinality reduces the number of possible transport strategies to only those in which some of the robots push other robots which in turn push the ob-



(a)

(b)

(c)

Figure 5: Graphs showing the performance of the best evolved groups of the *T-condition* during post-evaluation. Each graph shows the performance of homogeneous groups of 2, 3, and 4 robots while transporting object of different masses, as indicated on the x-axes. The performance is measured in term of distance the object has been transported in 60s trials. The object length is: (a) 20cm; (b) 30cm; and (c) 40cm. Each point in the boxes refer to the performance in a single trial.

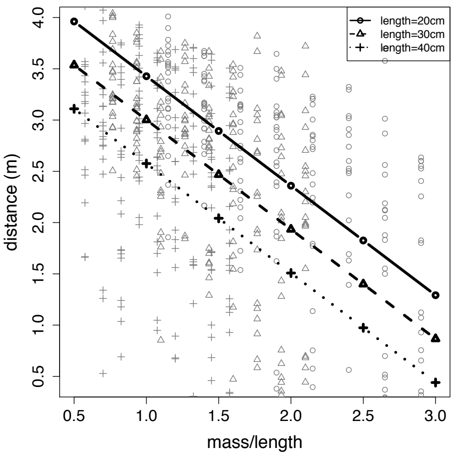


Figure 6: Graph showing the relationship between the different factors (i.e., length and ratio between mass and length of the object) and group performances (i.e., the distance the object has been moved in 60s trials) using linear mixed model. The graphs shows data points and fixed effect for different object length, with crosses for 40cm length objects, triangles for 30cm length objects, and circles for 20cm length objects.

ject. Since the robots usually need more time to arrange themselves in this transport formation, the performances in this test condition tend to be lower. There is also a slight loss of performance for all the four-robot groups, which is caused by a combination of factors. The most relevant is the effect of robot-robot collisions which are more likely to happen in larger groups. If during a collision the robots occlude each other camera, they have no means to distinguish whether they are in touch with another robot or with the object. Therefore, there are robot-robot collisions that trigger pushing behaviour that penalises the group. These are temporary deadlocks that tend to disappear after few seconds. However, these are events that increase the time for coordination and shorten the time for pushing. Moreover, in larger groups, for any starting positions, the robots generally need more time to arrange themselves in a way to exert the forces required to transport the object.

We have also run a statistical analysis on the performances of the best group of the *T-condition*, using linear mixed models, taking into account random effects of the starting positions. This analysis has shown that we can not reject the null hypothesis that there is not differences between two, three, and four-robot groups. In other words, group cardinality has no effects on performance. Statistically significant effects are those produced by the length and the ratio between mass and length of the object. The bigger the ratio, the lower the group performances; and the longer the object length the lower the group performances (see Fig. 6). The effect of mass/length ratio on performances is expected since even the lighter object is heavy enough to require the effort of all the robots of a group. Thus, heavier objects can only produce a loss of performance. The effect of length can be

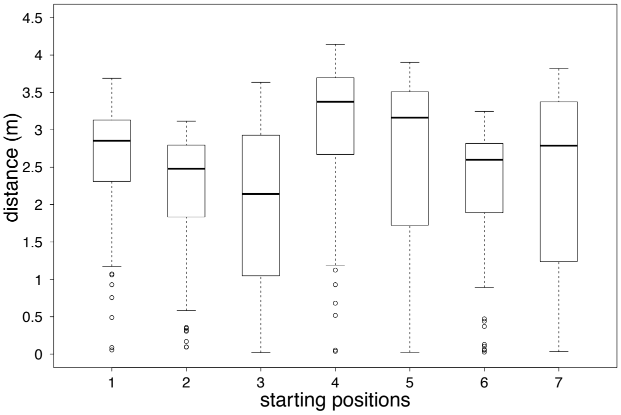


Figure 7: Graphs showing the performance of the best evolved solution of the *T-condition* during post-evaluation for each of the 7 different starting positions. Each point in the boxes refer to the performance in a single trial.

explained by taking into account the particular behaviour of robots of this group. Each robot, once in the proximity of the object, moves along the object perimeter by applying forces in different positions until an effective distribution of forces is found. This means that longer the object length, longer the time it takes to find an effective distribution of forces. Therefore, the loss of performance is a consequence of less time left for transport.

In Fig. 7, we show how the performance of the best evolved solution of *T-condition* varies with respect to the different robot initial positions. Starting position n. 3 is the one with the worst performance. Visual inspection of the robots behaviour have shown that this starting position penalises four-robot groups, since it generates a higher number of robot-robot collisions, some of which triggered the "maladaptive" pushing behaviour mentioned above. However, the graph shows that the groups manage to efficiently cope with the effects produced by the variability in starting positions.

Recruitment Behaviour

In this Section, we show results of a further post-evaluation test, called *recruitment-test*, which shows that the majority of the best evolved solutions can show a simple form of recruitment behaviour which is serendipitous, since it is not part of the behavioural repertoire selected for during evolution. We have post-evaluated each best solution of each run of the *T-condition* with a test in which two-robot groups have to transport a 20cm, 280g object. Each solution has been tested 21 times (i.e., 3 times the 7 starting positions illustrated in Fig. 3). In order to test the group for recruitment, we modified the operating conditions in the following. For each trial, the starting position of robot A is moved up to a distance of 80cm from the object. From its starting position robot A can not perceive the object. Moreover, robot A is initially "frozen". It can not move. Robot B starts the trials as shown in Fig. 3, at 50cm from the object. The camera view of robot B is extended to 100cm. Robot A is free to move if robot B gets closer than 10cm to it after having touched the object.

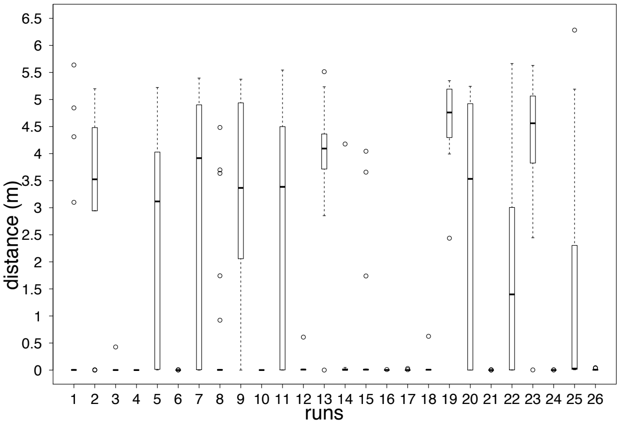


Figure 8: Graph showing the distance a 20cm, 280g object has been transported by the best evolved solutions for each run of the *T-condition* during post-evaluation in recruitment-test. Each point in the boxes refer to the performance in a single trial.

Recruitment happens if robot B finds out first that the object can not be move by pushing it. Then, it has to move towards robot A. Once robot B gets sufficiently close to robot A, the latter is free to move. However, robot A needs to be guided towards the object by robot B, since only robot B perceives the object from a distance longer than 50cm. In each trial, we infer whether or not recruitment took place simply by looking at the distance the object has been transported after 120s. If the object has not been moved at all, we assume that something went wrong with the recruitment. Otherwise, we assume that the recruitment has been successful.

The graph in Fig. 8 shows that the distance the object has been transported during the recruitment-test. The graph shows that 10 out of the 26 groups manage to transport the object more than 1 meter away from its initial positions in more than half of the trials (see Fig. 8, median values). Considering that both robots are required to transport the object, this result indicates that robots of these groups have developed an effective recruitment behaviour. This social behaviour is generated by the robots instinct to approach the group-mate after having found out that the object required more robots to be transported.

Conclusions

In this study, we have illustrated the results of a set of experiments in which a swarm of robots are required to transport an object by coordinated pushing actions. We have shown that, robots equipped with torque sensors can generate transport strategies that are fairly robust to variability in length and mass of the object, as well as to the cardinality of the group. The behaviour of the best evolved groups looks analogous to the one observed in some species of ants, where the coordination is accomplished through a set of individual responses including realignment, repositioning and reaction to stigmergic communication.

We observed that, in best evolved groups of the *T-condition*, the torque sensors is primarily used by the robots to generate feedback to distinguish between actions that results in a successful object movement from those that don't. This feedback helps the robots to recover from deadlocks and to coordinate the forces in order to avoid to work against each other. This is a clear example of how small changes into the sensory-motor configuration of the robot can help evolution to find successful evolutionary paths that lead to robust and adaptive solutions. We have also shown that our robots are capable of developing recruitment responses even if recruitment has not been considered (i.e., selected for) during evolution. Recruitment behaviour stems from a robot's tendency to approach the closest group mates. In normal conditions, this behaviour has the function to bring the robots on the same side of the object to avoid to work against each other during transport. However, if for any reason a robot is left alone pushing a too heavy object, the tendency to approach a group mate can take the robot away from the object closer to a distant robot, mimicking a simple form of recruitment. This suggests that ESR methods offer the opportunity for serendipitous behaviours that improves the adaptability of the system.

To validate the results of our study, ongoing work focuses on test with real robots. In the future, we aim to test the effectiveness of ESR design methods in generating solutions that are required to adapt to objects of irregular shapes. We also aim to expand the behavioural repertoire of the swarms, by integrating more complex form of recruitment and object retrieval responses that require the object to be transported to specific locations.

Acknowledgements

Muhanad H. Mohammed thanks Iraqi Ministry of Higher Education and Scientific Research for funding his PhD.

References

Beer, R. D. and Gallagher, J. C. (1992). Evolving dynamic neural networks for adaptive behavior. *Adaptive Behavior*, 1(1):91–122.

Berman, S., Lindsey, Q., Sakar, M. S., Kumar, V., and Pratt, S. C. (2011). Experimental study and modeling of group retrieval in ants as an approach to collective transport in swarm robotic systems. *Proceedings of the IEEE*, 99(9):1470–1481.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(2):1–41.

Doncieux, S. and Mouret, J.-B. (2014). Beyond black-box optimization. *Evolutionary Intelligence*, 7(2):71–93.

Dorigo, M. and Şahin, E. (2004). Guest editorial. Special issue: Swarm robotics. *Aut. Rob.*, 17(2–3):111–113.

Dorigo, M. and et al. (2013). Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. *Robotics & Automation Magazine, IEEE*, 20(4):60–71.

Francesca, G. and et al. (2014). An experiment in automatic design of robot swarms. In *Proc. of the* 9^{th} *Int. Conf. on Swarm Intelligence*, pages 25–37. Springer.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA.

Groß, R. and Dorigo, M. (2004). Cooperative transport of objects of different shapes and sizes. In *Ant colony optimization and swarm intelligence*, pages 106–117. Springer.

Groß, R. and Dorigo, M. (2008). Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling. *Adaptive Behavior*, 16(5):285–305.

Groß, R. and Dorigo, M. (2009). Towards group transport by swarms of robots. *International Journal of Bio-Inspired Computation*, 1(1):1–13.

Jakobi, N., Husbands, P., and Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in artificial life*, pages 704–720. Springer.

Kube, C. R. and Bonabeau, E. (2000). Cooperative transport by ants and robots. *Robotics and autonomous systems*, 30(1):85–101.

Kube, C. R. and Zhang, H. (1993). Collective robotics: From social insects to robots. *Adaptive Behavior*, 2(2):189–218.

Mataric, M. J., Nilsson, M., and Simsarin, K. T. (1995). Cooperative multi-robot box-pushing. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 556–561.

Michel, O. (2004). Webotstm: Professional mobile robot simulation. Available at http://arxiv.org/abs/cs/0412052. cs/0412052.

Mondada, F. and et al. (2009). The e-puck, a robot designed for education in engineering. In *Proc. of the* 9^{th} *Int. Conf. on Autonomous Robot Systems and Competitions*, volume 1, pages 59–65.

Sudd, J. H. (1965). The transport of prey by ants. *Behaviour*, 25(3/4):234–271.

Trianni, V. (2014). Evolutionary robotics: model or design. *Frontiers in Robotics and AI*, 1:1–6.

Wang, Z.-D., Takano, Y., Hirata, Y., and Kosuge, K. (2004). A pushing leader based decentralized control method for cooperative object transportation. In *Proc. of IEEE/RSJ International Conference Intelligent Robots and Systems (IROS)*, volume 1, pages 1035–1040. IEEE.

Closed-loop acquisition of behaviour on the Sphero robot

Oswald Berthold* and Verena V. Hafner*

*Adaptive Systems Group, Dept. of Computer Science, Humboldt-Universität zu Berlin
{bertolos|hafner}@informatik.hu-berlin.de

Abstract

In this paper, we investigate the closed-loop acquisition of basic behaviours on Sphero – a real spherical robot. We impose the additional requirement of learning from scratch in a single episode. The behaviour is encoded as an inverse model for stabilization and sensory target tracking tasks using recurrent neural networks.

Keywords: self-exploration, embodiment, online closed-loop learning, goal-babbling, reservoir computing

Introduction

If a robot has not been preprogrammed on how to execute a given motion, one of the fundamental things it has to accomplish is to acquire coordinated control over its body which is a prerequisite for locomotion, foraging and other more complex behaviours. We approach the problem by considering how a robot can acquire these basic skills through self-exploration.

Controlling the motion and behaviour of a physical system requires to invert the system which is usually accomplished by providing a controller (inverse model) for the given system. In simple systems, inverting means finding the correct sign and magnitude of the sensory response to motor output. Here we want to simultaneously learn and exploit such a model by using sensor, motor and reward samples obtained incrementally from the system. Each sample is then used for adapting the weights of a linear combination of states that provides the motor signal.

We restrict our investigation to a very simple and low Degree-of-Freedom (DoF) system and a fully observable environment. Instead, we put the emphasis on the dynamic control scenario, coping with the state dynamics, and the initial transient of the learning curve. By that we mean the amount of learning steps until the error measure goes below a given threshold.

A perfect robot that complies with these restrictions is the Sphero [1], a small spherical robot with a diameter of about 6 cm and a shock-resistant and waterproof shell of polycarbonate, see Figure 1. It can be operated using the Robot Operating System (ROS) [2] [3] which allows sending commands and receiving the sensor stream.

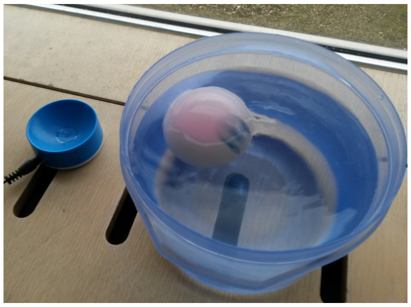


Figure 1: Sphero robot learning to move while suspended in water.

In addition, we are only considering bootstrapping in the sense of learning the inverse model from scratch. This means that in a learning run the system starts to operate with zero output, to which exploratory action is added and, starting with the first sample, to alter the model and modulate the exploration. We consider this type of basic control as an initial step in a bottom up strategy for building adaptive systems. In contrast to a traditional control theoretic approach that requires specification of a plant model at design time, our method aims at more generally learning any such model during interaction with the world.

This basic problem still comes in many varieties and is the subject of several areas of research. Depending upon the area, different aspects are emphasized. In adaptive control (Astrom and Wittenmark, 1994), for example, usually the parameters of a structurally fixed model are adapted to slowly varying variables. In biological sensorimotor learning the problem is mostly considered as how the human nervous system manages to identify and acquire or adapt to new or changing circumstances (Wolpert et al., 2011). For Em-

[1] http://www.gosphero.com/sphero/

[2] http://www.ros.org/

[3] https://github.com/mmwise/sphero_ros

bodied Artificial Organisms the goal is to design and learn controllers that exploit the information transforming properties of the physical body and its interaction with the environment (Iida et al., 2004). Finally, in classical Reinforcement Learning (RL) a lot of focus is put on handling potentially sparse and temporally distant rewards, episodic learning and theoretical convergence bounds (Sutton and Barto, 1998).

While online learning is generally biologically more plausible than batch updating, comparing our approach in detail to the way animals learn during their development is beyond the scope of this work. Animals do not, in many cases, actually bootstrap but instead learn on top of an evolutionary prior which might have evolved for learnability in higher animals. The learning model we consider is controlled by a set of hyperparameters that, by definition, are constant or change only slowly, during any given learning episode. The artificial substitute for a biological evolutionary prior can be thought of as a good set of hyperparameters and some basic pre-wiring that lets the system learn effectively with stability under the usual environmental conditions.

We will review related work in more detail in the next section, present the abstract learner in the Model section and then discuss a series of experiments empirically investigating the learning performance and the behavioural generalization of the acquired models on the Sphero robot. We conclude by summarizing and giving an outlook on ongoing and future work.

Online motor learning

The online sensorimotor learning problem has been treated in different fields such as Reinforcement Learning, Computational Neuroscience or bio-inspired robotics. In our review we focus on the latter two as they are much more relevant to our approach.

In computational neuroscience all learning must eventually be formulated as correlational learning, following the Hebbian postulate. Since this type of learning is self-amplifying, mean removal from the two Hebbian factors (Sejnowski et al., 1989; Porr and Wörgötter, 2003; Wörgötter and Porr, 2005), as well as a third modulatory factor has been introduced into the update rule as a way to provide stability (Porr and Wörgötter, 2007; Hoerzer et al., 2012). In addition, these differential three-factor rules could be shown to be equivalent to Temporal Difference (TD) learning in RL (Kolodziejski et al., 2008).

A more high-level approach is that of explicitly learning forward inverse model pairs via sampling of the sensorimotor space and exploiting the forward model part for inferring the Value of the expected consequences under the corresponding inverse model taking control. This method is conceptually simple and attractive but deploying it in an online manner presents some difficulties (stability / plasticity dilemma) (Wolpert and Kawato, 1998), (Wolpert et al., 2011), (Demiris and Dearden, 2005), (Schillaci and Hafner, 2011). Nonetheless, successful experiments in this direction are presented in (Schillaci et al., 2014), which integrates the multi-modal sensing problem, or in (Stoelen et al., 2012) for a trajectory tracking task and associations of labels and low-level sensory input.

A modification of inverse forward model pair learning through exhaustive sampling that scales to higher-dimensional sensorimotor spaces is online goal-directed exploration (goal-babbling). Here the agent is sampling from data generated through self-exploration while it is trying to reach a goal (Rolf et al., 2011). This is done using kinematic control. Similar experiments on learning high-dimensional control tasks are presented in (Baranes and Oudeyer, 2013) using an intrinsic motivation approach.

For completeness direct learning methods in RL need to be mentioned. These have been applied with success to high-DoF systems using for example Dynamic Movement Primitives. A good survey of these methods and their application in robotics can be found in (Kober and Peters, 2012). We do not consider these methods any further here, as their updates are usually done over entire episodes. This is harder to reconcile with continuous (correlational) biological plasticity mechanisms.

A type of neural network architecture suitable for dynamical system control and online learning is known as Reservoir Computing (RC). The term comes from the interpretation of the recurrent hidden layer as an excitable computational reservoir. It is a unification of computational models known as Echo State Networks (ESN) (Jaeger, 2001), or Liquid State Machines (LSM) (Maass et al., 2002). The differences between those are based on the underlying neuron models. The reservoir architecture provides two computational features: a high-dimensional non-linear expansion of the original input space and memory of recent input trajectories through the recurrent connections. Our experiments rely mostly on the first property.

Considering the reservoir model of (Hoerzer et al., 2012) we note that it can be regarded as a variant of the Continuous Actor-Critic Learning Automaton (CACLA) proposed in (Hasselt, 2012), which is an algorithm rooted in classical RL. Similar to the results of (Kolodziejski et al., 2008) on the relation of correlation and TD learning, this provides a link on an algorithmic level between low-level neural network based learning models and an RL-based Temporal Difference formulation. For the work presented here we extend the model by including a physical device in the learning loop. This has the main effect that the sensorimotor delay, that is, the delay between a motor command and its sensory consequence cannot be assumed to be exactly equal to one but might be larger. This fact has to be considered in the formulation of the learning rule.

Model

Our scenario consists of a robot, an environment and a neural control circuit. The robot we consider here is the Sphero, the environment consist of the transformation laws describing how new sensor states are computed from a motor command, and the neural circuit is a reservoir network. It consists of an input layer, a single hidden layer of recurrently connected neurons, and a linear output layer referred to as *readouts*.

Reservoirs are most often trained with batches of supervised training sets (Lukoševičius and Jaeger, 2009). Here, instead, we employ incremental updates with a Hebbian learning rule. The learning rule is modulated by a performance measure defined on sensor states. We assign problem specific performance measures, in this case for example the negative quadratic distance to an externally provided target value. The learning task is to invert the robot-environment coupling to find the motor command which generates a sensor state representing good performance.

The inverse model is realized as a function $\boldsymbol{y} = f(\boldsymbol{u}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{W}^{\text{out}})$ mapping sensory inputs $\boldsymbol{u} \in \mathbb{R}^n$ to motor outputs $\boldsymbol{y} \in \mathbb{R}^m$ with n the sensor dimension and m the motor dimension, $\boldsymbol{x} \in \mathbb{R}^N$ is a hidden state with dimension $N >> n$. The hidden state is both driven by the sensory input and recurring upon itself via connection strengths drawn randomly once at the beginning of the experiment. The motor output $\boldsymbol{y}$ is a linear combination of the hidden state with weights $\boldsymbol{W}^{\text{out}}$. We realize the function f as a reservoir.

Exploiting the universal modelling properties of reservoir networks, the task is reduced to finding parameters $\boldsymbol{W}^{\text{out}}$ that implicitly encode the inverse model. Without explicitly computing the performance gradient with respect to the parameters, we update the weights with a Hebbian rule which is modulated by a third factor. This factor is a binary indicator of recent improvements in performance. The accumulated weight changes reinforce successful actions (postsynaptic activation) for corresponding hidden states (presynaptic activation) when the reward signal is non-zero. It amplifies the correlation between the rewards generated by an exploration signal and the pre- and postsynaptic states (Hoerzer et al., 2012) such that states which yield reward are made more likely to occur.

The reason we use this type of learning is that there is no target for the motor signal. Conventional reservoir training assumes the existence of a supervised training set in terms of input and output pairs but here the target is only given indirectly. The output target has to be discovered by the learner through exploration.

The function f is realized in the following way by a reservoir. The hidden network state evolves according to

$$\triangle \boldsymbol{x}_t = \lambda \boldsymbol{W}^{\text{res}} \boldsymbol{r}_t + \boldsymbol{W}^{\text{in}} \boldsymbol{u}_t + \boldsymbol{W}^{\text{bias}} 1 \tag{1}$$

$$\boldsymbol{x}_{t+1} = (1-\tau)\boldsymbol{x}_t + \tau \triangle \boldsymbol{x}_t \tag{2}$$

$$\boldsymbol{r}_{t+1} = \tanh(\boldsymbol{x}_{t+1}) + \nu_{\text{state}} \tag{3}$$

The matrices $\boldsymbol{W}^{\text{res}}, \boldsymbol{W}^{\text{in}}, \boldsymbol{W}^{\text{bias}}$ are the $N \times N$ reservoir, $N \times n$ input, and $N \times 1$ bias matrices respectively. We use a reservoir size of $N = 500$, and input dimension $n = 4$. The scalar $\lambda = 0.99$ is a scaling factor to effect a desired spectral radius for the reservoir matrix. The connection probability for the reservoir matrix $\boldsymbol{W}^{\text{res}}$ is controlled by parameter $p = 0.1$. This means, we generate the matrix with sparse non-zero entries of density p. The non-zero entries themselves are drawn from a standard normal distribution. Then the matrix is rescaled to the given spectral radius λ. The state noise ν_{state} is uniformly distributed with limits -0.02 and 0.02 and is used as a regularizer. The network outputs are computed as

$$\boldsymbol{y} = \boldsymbol{W}^{\text{out}} \boldsymbol{r} \tag{4}$$

At this stage white Gaussian noise $\boldsymbol{\nu}$ is added to the output $\boldsymbol{y}$ yielding

$$\hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\nu} \tag{5}$$

This is the final stage motor signal before it is sent to the actuators.

Performance measure and learning rule

We designate the measure of the current performance of the network with P. It depends on the motor output k time-steps in the past. Here, we mostly use the negative squared error with respect to an externally imposed target such as $P_i = -(u_i - \text{target}_i)^2$ for a given sensory input i or the sum $P = -\sum_i (u_i - \text{target}_i)^2$. A low-pass filtered version $\bar{P}_t = \alpha P_{t-1} + (1-\alpha) P_t$ is also maintained with $\alpha = 0.8$. The modulator signal is derived as an approximation of the performance derivative from P_t and $\bar{P}_t$ via

$$M_t = \begin{cases} 1 & \text{if } P_t > \bar{P}_t \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

and the weight update then is

$$\Delta \boldsymbol{W}^{\text{out}}_{i,t} = \eta_{i,t} \boldsymbol{r}_{t-k} (y_{i,t-k} - \bar{y}_{i,t-k}) M_t \tag{7}$$

with $\bar{y}_{i,t}$ being a low-pass filtered version of $y_{i,t}$ analogously to $\bar{P}$. We use a time-dependent learning rate η_t with a half-time on the order of 1000 time steps. Also, we apply soft weight-bounding to avoid run-away solutions for the output weight vector $\boldsymbol{W}^{\text{out}}$. The weight bounding is an additional multiplicative term in the update rule, which throttles the learning rate to zero if the norm of the weight vector comes close to an empirically determined threshold. Note that the standard Hebbian terms are indexed with $t - k$ whereas the modulator refers to the current time. The variable k is the sensorimotor delay which needs to be determined either through knowledge of the system or empirically. In the latter case this can be done using cross-correlation analysis. A graphical representation of the algorithm is displayed in Figure 2.

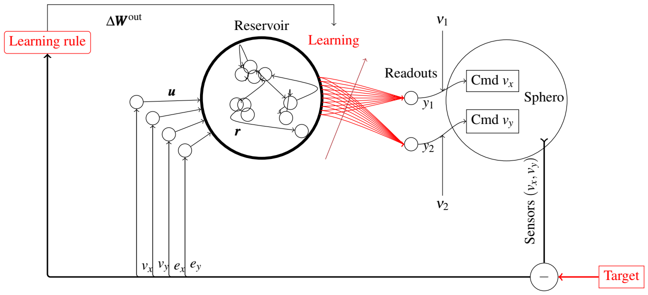


Figure 2: Graphical representation of the learning algorithm. The thick circle labeled Reservoir implements Eqs. 1,2 and 3, the red bundle of arrows and neurons y_1 and y_2 correspond to Eq. 4. After that, noise ν_1 and ν_2 are added and sent to Sphero's control input (Eq. 5). The red box "Learning rule" contains both Eqs. 6 and 7. The boxes labelled "Cmd" also contain the output scaling factor gain$_{\text{out}}$ which is specific and usually constant for a given robot. The variables $v_{x,y}$ and $e_{x,y}$ are the measured velocity and velocity errors respectively.

Experiments

Sphero

We use the Sphero robot [4] in these experiments. The robot is shown in Figure 1. All the experiments were performed with the Sphero suspended in a bucket of water. The robot skin is a shock-resistant and water-proof spherical shell of polycarbonate. Inside there is a two-wheeled cart with differential drive and a vertical beam which pushes against the wall of the shell thus improving traction of the wheels. The cart has an onboard IMU which is used for path integration in order to maintain state and odometry. We use the ROS driver together with Python to implement our model and control the robot. In all the experiments we run the control loop at 20 Hz which results in a sensorimotor delay of four time-steps, so $k = 4$. In Figure 3 we provide Sphero's sensorimotor transfer curve where we sweep twice through the full motor range and record the corresponding velocity measurements. The curve is quasi linear with a slight hysteresis that results from the underlying control system. Around zero we observe some discontinuity due to Sphero's internal cart needing to turn around.

The robot is freely moving within the constrained space of the bucket. When it is floating freely it will accumulate some linear velocity and move toward the rim of the bucket. On contact with the rim Sphero slightly rebounds and slowly turns due to the accumulated noise in the odometry. None of these interactions are handled specifically in our setup, they just contribute to the overall noise. The error signals in the experiments are always only relative to onboard (idiothetic) odometry, assuming that improving the state estimation, for example by reference to allothetic cues, will not affect the velocity regulation, as these two processes take place on different time scales.

Before applying the network output to the Sphero control system it is scaled by a constant, in our case gain$_{\text{out}}$ $= 100$. This means the network output must go from -2.5 to 2.5 in order to cover the full motor range of the robot. This is the same in all experiments. The standard ROS interface for the Sphero accepts a cartesian velocity input. This input specifies a two-component velocity vector which is the set-point for the underlying control system. Internally, it is converted into a single forward velocity and a heading of the cart inside the shell.

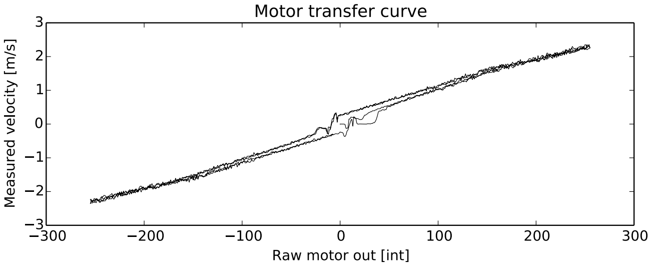


Figure 3: Sphero's motor transfer curve. The Sphero's stabilized control system accepts inputs in the range [0, 255] with an additional direction bit, mapped together from -255 to 255. The velocity is measured in m/s on the outer shell surface. Here we plot measured velocity response over motor output for two sweeps of the motor range. The response is quasi-linear with a slight hysteresis.

Constant target

Considering that the Sphero can physically move in the plane, it can at most move along two axes (Degrees of Freedom) with respect to the odometry's frame of reference. In the simplest conceivable scenario the network is trained to regulate to a constant velocity on a single axis of motion of the robot. At the beginning of a run, a random target velocity value v_{target} is drawn from a uniform distribution such that $v_{\text{target}} \in [0.5, 1.2]$. Additionally the sign of the target is assigned randomly. The learning task is to find output weights which result in a readout signal of the correct magnitude and sign such that the resulting measured velocity is close to the target. The averaged squared error for 10 runs is plotted over the entire run-length of 5000 time-steps in Figure 4.

[4] http://www.gosphero.com/sphero/

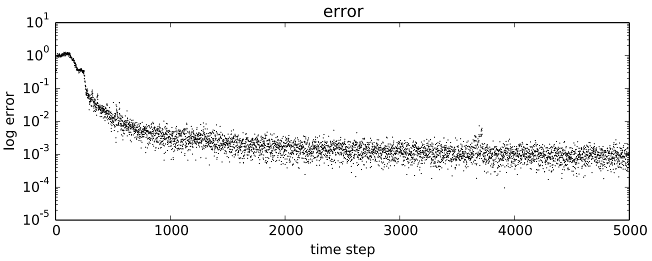


Figure 4: Log squared error curve averaged over 10 runs for the learning task of regulating the measured rolling velocity along a single body axis to a given constant target. After 500 time-steps the error is reduced by two orders of magnitude. That is 25 seconds of real-time at a loop frequency of 20 Hz.

Jumping target

Motivated by results from earlier experiments on a simulated system that indicate improved robustness when training against a moving target, we let the Sphero learn to stabilize a target rolling velocity which changes during the learning episode. The evolution of the log squared error is shown in Figures 5 and 6 for two cases. In the first case of Figure 5 the target jump is only in magnitude but maintains the same sign while in the second case of Figure 6, the jump always changes the sign. The jumping targets are visible in the error plot as pronounced spikes in the curve.

As a test of the generalization of the acquired inverse models, we evaluate them against a different target sequence. As can already be seen in the error plots, the error for the constant target case goes down to a much lower value than that for the jumping target case. This is due to large jumps in the output weights after the occurrence of a target jump during learning. This leads to premature saturation of the weight vector and consequent stalling of the plasticity. The lower error performance is also reflected in the case of testing the network trained for a constant target against the jumping target, which also generalizes to the inverse sign without having seen this situation during training as shown in Figure 7.

Sphero in the plane

Having obtained some basic results with the Sphero for motion along a single body axis, we now extend the setup to two dimensions. Internally, Sphero is controlled by a forward velocity and a heading. Cartesian velocity vectors are converted by the underlying control system. In order to have two readouts of the same "kind", we choose to use the cartesian control interface. Again for simplicity we use the sum of the two error components as a single scalar performance measure. Thus, although the two readout units both share the presynaptic state and the modulator signal, the system is still able to achieve adequate performance in comparison with the one-dimensional case. Here, we only show the learning curve for the constant two-dimensional target case in Figure 8.

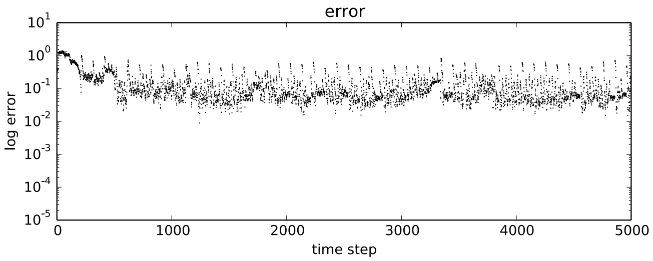


Figure 5: Log squared error curve averaged over 10 runs for learning to regulate the measured rolling velocity to a changing target that keeps the same sign and thus avoids the discontinuities around motor values of zero when the ball turns around.

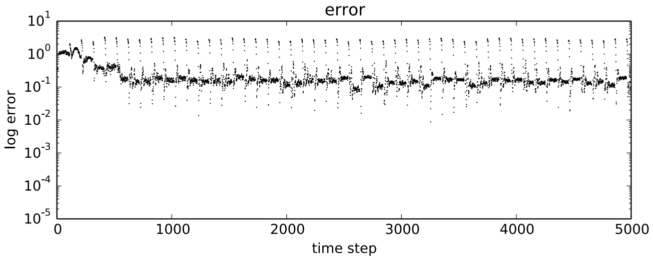


Figure 6: Log squared error curve averaged over 10 runs for learning to regulate the measured rolling velocity to a target changing both magnitude and sign. In this case learning saturates even faster than in the magnitude only condition above.

We also perform the same evaluation of behavioural generalization of the acquired models by evaluating a network trained for constant targets against one trained for changing targets. While the time-series of the regulated velocity is different, the average error over the target change episodes is on the same order of magnitude for both cases, see Figure 9. The tracking of the target is not nearly as good as for the one-dimensional case indicating the necessity for tuning the learning setup to this more complex scenario.

Conclusion

We have presented a reservoir computing based approach to online motor learning, that is, the acquisition of inverse models from scratch for simple motion behaviours on a low-dimensional physical robot, the Sphero.

The proposed learner is able to quickly establish an adequate motor response based on sensory input in order to regulate the input to externally generated target values. The high-level input we require is a monotone performance measure that can be defined on the observed sensors. It has to be seen, whether monotonicity can be warranted for more

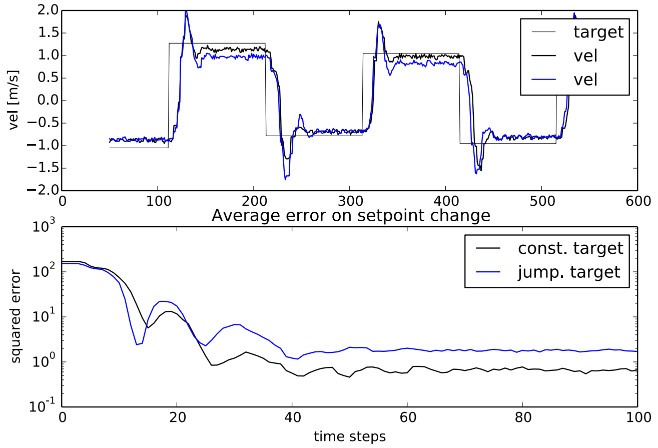


Figure 7: Evaluating the generalization: here we train two networks, one with a constant target (black) and one with a jumping target (blue). We test both with the same randomly jumping target (thin black) with alternating sign. The top graph shows the timeseries of the target and the measured controlled velocity responses for both networks. The bottom graph shows the log squared error averaged over 50 jumps.

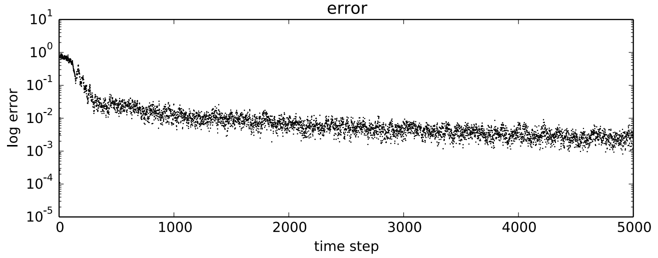


Figure 8: Log squared scalar error curve for learning to regulate the measured rolling velocity to a two-dimensional target. Again, the error is decreasing by two orders of magnitude after about 500 time-steps. It can also be seen that learning has not fully converged in this case before termination of the learning run.

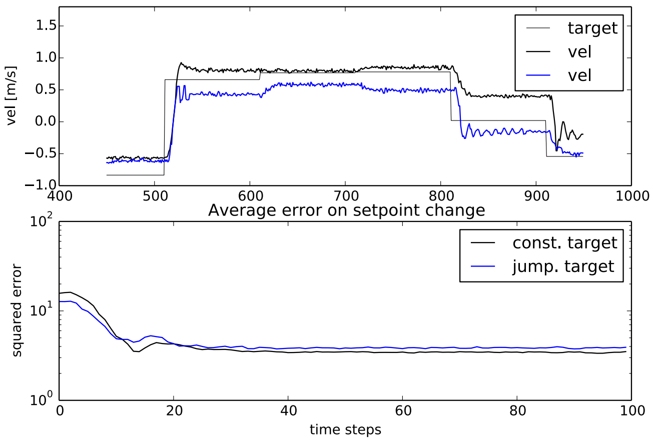


Figure 9: Again we evaluate the generalization of a network trained for a constant target (black) vs. one trained for a jumping target (blue). The top graph shows the timeseries of one component of the two-dimensional response for each network to the same target (thin black). The bottom graph shows the log squared error averaged over 12 jumps.

complex behaviours. Failing that, another approach would be a hierarchical architecture. We have demonstrated that the approach yields useful results for the cases of one- and two-dimensional motion of a real spherical robot. Even in cases of training only for a single stationary sensory target, the resulting network manages proper control of the physical system for changing targets. We presume that the spatial richness of the state expansion of the reservoir together with performance measure's monotonicity is an important ingredient for the speed of learning. This is suggested by preliminary experiments on reducing the reservoir size and comparison with similar algorithms using different types of neural network architectures. We would like to point out that the error numbers obtained in the experiments still include the learners intrinsic exploratory noise signal. Clamping the noise to zero further improves the performance. Further investigation of these latent issues has to be left to future work.

Another important aspect is sensitivity of the learner to the hyperparameters. It is known from the neural network literature that input scaling is one such important parameter. A systematic mapping of the learning performance in the hyperparameter space for a comparable model system is in preparation. This also applies to investigating the dependence of the performance on the noise levels present in the system.

We would like to thank Helmut Hauser for helpful comments on an earlier draft of this paper and our colleagues in the Adaptive Systems Group for discussions.

References

Astrom, K. J. and Wittenmark, B. (1994). *Adaptive Control*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition.

Baranes, A. and Oudeyer, P.-Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1):49–73.

Demiris, Y. and Dearden, A. (2005). From motor babbling to hierarchical learning by imitation: a robot developmental pathway.

Hasselt, H. v. (2012). Reinforcement learning in continuous state and action spaces. In Wiering, M. and van Otterlo, M., editors, *Reinforcement Learning*, volume 12, chapter Adaptation, Learning, and Optimization, pages 207–251. Springer Berlin Heidelberg.

Hoerzer, G. M., Legenstein, R., and Maass, W. (2012). Emergence of complex computational structures from chaotic neural networks through reward-modulated hebbian learning. *Cerebral Cortex*.

Iida, F., Pfeifer, R., Steels, L., and Kuniyoshi, Y., editors (2004). *Embodied Artificial Intelligence, International Seminar, Dagstuhl Castle, Germany, July 7-11, 2003, Revised Papers*, volume 3139 of *Lecture Notes in Computer Science*. Springer.

Jaeger, H. (2001). The ”echo state” approach to analysing and training recurrent neural networks - with an erratum note. Technical report.

Kober, J. and Peters, J. (2012). Reinforcement learning in robotics: A survey. In Wiering, M. and Otterlo, M., editors, *Reinforcement Learning*, volume 12, chapter Adaptation, Learning, and Optimization, pages 579–610. Springer Berlin Heidelberg.

Kolodziejski, C., Porr, B., Tamosiunaite, M., and Wörgötter, F. (2008). On the asymptotic equivalence between differential hebbian and temporal difference learning using a local third factor. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 21*, pages 857–864. Curran Associates, Inc.

Lukoševičius, M. and Jaeger, H. (2009). Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127–149.

Maass, W., Natschläger, T., and Markram, H. (2002). Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560.

Porr, B. and Wörgötter, F. (2003). Isotropic sequence order learning. *Neural Computation*, 15(4):831–864.

Porr, B. and Wörgötter, F. (2007). Learning with "relevance": using a third factor to stabilize hebbian learning. *Neural Comput*, 19(10):2694–719.

Rolf, M., Steil, J. J., and Gienger, M. (2011). Online goal babbling for rapid bootstrapping of inverse models in high dimensions. In *Development and Learning (ICDL), 2011 IEEE International Conference on*, volume 2, pages 1–8.

Schillaci, G. and Hafner, V. V. (2011). Random movement strategies in self-exploration for a humanoid robot. In *Proceedings of the 6th international conference on Human-robot interaction*, pages 245–246.

Schillaci, G., Hafner, V. V., and Lara, B. (2014). Online learning of visuo-motor coordination in a humanoid robot. a biologically inspired model. In *Proceedings of The Fourth Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics*.

Sejnowski, T., Chattarji, S., and Stanton, P. (1989). Induction of synaptic plasticity by hebbian covariance in the hippocampus. chapter The Computing Neuron, pages 105–124. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Stoelen, M. F., Bonsignorio, F., Balaguer, C., Marocco, D., and Cangelosi, A. (2012). Online learning of sensorimotor interactions using a neural network with time-delayed inputs. In *Development and Learning and Epigenetic Robotics (ICDL), 2012 IEEE International Conference on*, pages 1–6.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition.

Wolpert, D. M., Diedrichsen, J., and Flanagan, J. R. (2011). Principles of sensorimotor learning. *Nature Reviews Neuroscience*, 12(12):739–751.

Wolpert, D. M. and Kawato, M. (1998). Multiple paired forward and inverse models for motor control. *Neural Networks*, 11(7):1317–1329.

Wörgötter, F. and Porr, B. (2005). Temporal sequence learning, prediction, and control: A review of different models and their relation to biological mechanisms. *Neural Computation*, 17(2):245–319.

How to play the Syntax Game

Luc Steels[1,2] and Emilia Garcia-Casademont[2]
[1]ICREA
[2] Institut de Biologia Evolutiva (UPF-CSIC)
Dr. Aiguader 88, Barcelona 08003, Spain

Abstract

This paper introduces the Syntax Game, a language game for exploring the origins of syntactic structure, specifically phrase structure. We define the game and propose a particular strategy for playing it. We show that this strategy leads to the emergence of a phrase structure grammar through the collective invention, adoption, and alignment of culturally established conventions.

Introduction

The topic of language evolution has been one of the many exciting research threads in Artificial Life since its beginning in the nineties. Most research so far has focused on the self-organization of vocabularies. The Naming Game, first published in the Alife journal in 1995 (Steels, 1995), emerged as the main model system, playing a similar role as the Prisoner's dilemma game for studying the origins of social cooperation. Many researchers have proposed strategies for playing the Naming Game, and studied the semiotic dynamics that unfolds, given particular strategies (Loreto et al., 2011). The Naming Game was also generalized to allow multiple words or deal with combinations of categories and there have been further experiments simulating the coevolution of concepts and names and to implement Naming Games on real robots (Steels and Hild, 2012).

But human languages go far beyond words. They feature sophisticated grammars which have two functions: (i) to express additional information beyond individual words, e.g. information about tense, aspect, modality, determination, information structure (foreground/background), spatial perspective, etc. and (ii) to help listeners avoid combinatorial explosions and ambiguity, both for semantic interpretation, where combinatorial search and ambiguity unavoidably arise when multiple words are used without signalling how these words are semantically related, and for parsing, because words or patterns tend to have multiple possible functions, generating combinatorial search. In this paper we focus on (ii), i.e. on how grammar arises to dampen syntactic and semantic ambiguity and avoid combinatorial search.

Although there have been several suggestions and proposals to create language game models for the self-organization of syntax, we are lacking a clear game, similar to the Naming Game, that all researchers could use to devise, test and compare strategies leading to grammar and study the semiotic dynamics these strategies generate. The primary goal of this paper is to propose such a game. It is called the Syntax Game. The Syntax Game is very similar to the Naming Game. The key difference is that agents now need to convey semantic networks involving multiple objects, instead of one or more categories pertaining to a single object, and that they can use syntactic means, such as word order, to convey how the arguments of different predicates in the network relate to each other.

Section 2 introduces the game itself and section 3 and 4 discuss why grammar is needed. Section 5 introduces a possible strategy for playing the game and section 6 shows results of simulation experiments with this strategy.

The Syntax Game

A language game models the interaction between two individuals of the same language community. The Syntax Game is a game of reference similar to the Naming Game, i.e. the speaker tries to draw the attention of the hearer to an object in the world. Both agents are assumed to maintain a model of the current situation, called a world model, through perception and action (Spranger et al., 2012). In computer simulations, this world model is synthesized based on an ontology of possible predicates. The Syntax Game then involves the following steps:

1. The speaker selects an object from his world model to act as the topic.
2. The speaker chooses what meaning distinctively describes this object and uses his own lexicon and grammar to translate this meaning into an utterance. The utterance is transmitted to the speaker.
3. The hearer parses this utterance using his own lexicon and grammar in order to reconstruct a possible meaning.
4. The hearer interprets this meaning in terms of his own world model in order to find out what topic the speaker intended. The hearer then signals to the speaker which object he interpreted.

5. The speaker signals success if the topic identified by the hearer is the same as the topic originally chosen by the speaker. If they differ, the speaker signals failure but also points to the object he originally chose.
6. Both speaker and hearer then expand and align their lexicons and grammars based on the outcome of the game.

In classical Naming Game research, the topic is a single object, the meaning a single category, e.g. a color, and the utterance a single word. The Syntax Game allows meanings with several categories and relations implicating several objects, and the utterance can be a set of sequentially ordered words organized in phrases. The challenge of the Syntax Game is to find a language strategy that solves this problem, in other words, that leads to the self-organization of a shared grammar in the population which allows the agents to have communicative success while minimizing the effort in semantic interpretation and syntactic parsing.

Reducing semantic uncertainty

Let us first introduce the sort of meanings speakers and hearers should be able to express. A *world model* consists of a set of objects corresponding to real world entities, e.g. a ball, a pyramid, a person. The objects are labeled as *o-1*, *o-2*, etc. The world model furthermore consists of a conjunction of facts which are true for the current world state, e.g. that the ball is moving, the pyramid is red, and the person is pushing the ball towards the pyramid. World models are represented using standard First Order Logic, that means in terms of predicates with arguments. Predicates are decomposed into an attribute and a value, such as color/red or material/plastic and written in prefix notation. The distinction between attributes and values is needed to generate semantic categories to be used by the grammar (as explained later). Unary predicates are written down as

(attribute value object)

as in

(color red o-1) or *(material plastic o-1)*

N-ary predicates, which represent relations, are decomposed. For example a predicate, *moving-away-from* with two arguments, for a mover and for the object the mover moves away from, is decomposed into three predicates, as in the following example:

(moving away o-3)
(mover-moving-away o-3 o-1)
(moving-away-from o-3 o-2)

There is one predicate *moving* with the value *away* for the relation itself which thus becomes reified as *o-3*. This reification is needed because the relation can also be a topic. For example, if the speaker says “the ball moves to the block”, the referent of the whole sentence is the moving relation. The other predicates (*mover-moving-away* and *moving-away-from*) explicitly introduce the arguments of the relation. The attribute is the name of the argument, the value is the relation itself, i.e. *o-3*, and the object being predicated is the filler of the argument, e.g. *o-1* in the case of the mover. This decomposition has a number of advantages, and is quite common in AI representations. One advantage is that all facts are tuples with 3 elements: the attribute, the value, and the predicated object.

The different facts in the world model form a *semantic network*. The nodes in the network are facts and if an object occurs more than once in different facts a link is established between them. For example, Figure 1 represents an event where a small paper moves away from a wooden table.

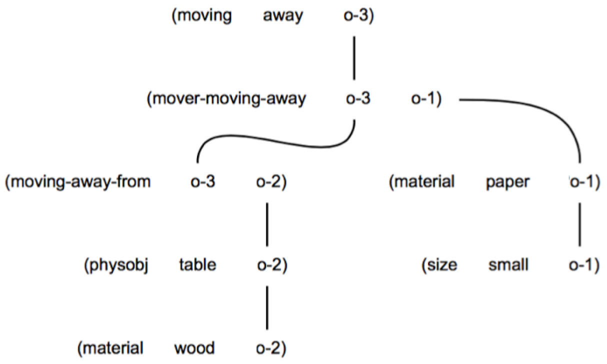


Figure 1: Example of a semantic network representing the world model of a speaker or a hearer.

The speaker in the Syntax Game chooses one object in such a network as the topic (for example, the wooden table *o-2* or the moving event *o-3*) and selects a subnetwork as the meaning of his utterance. This subnetwork can have varying degrees of complexity depending how many properties and relations are involved. For example for the world-model in Figure 1 we could have:

1. *The table* (topic = o-2)
2. *The wooden table* (topic = o-2)
3. *The paper moving away from the table* (topic = o-1)
4. *(I want) the small paper moving away* (topic = o-3), etc.

The speaker knows which specific objects are involved (o-1, o-2, etc.) but the hearer does not. A word like “wooden” signals that there is a wooden object in the scene but not which object is intended. “Table” introduces another object but we do not know whether it is the same as the one introduced by “wooden”, because there could be another wooden object, e.g. a block on the table. So after performing lexicon-lookup, the hearer can only derive a set of disconnected semantic subnetworks where the arguments are variables as opposed to objects (Figure 2).

Variables are written with a question-mark in front, as in *?o-1*, *?o-2*, etc. Note also that a fact for the topic has been added. The attribute is called *topic* and the value is either speaker or hearer. The argument of the predicate is the object chosen as the topic, which is here still a variable, namely ?o-8.

Semantic interpretation consists in finding bindings for all

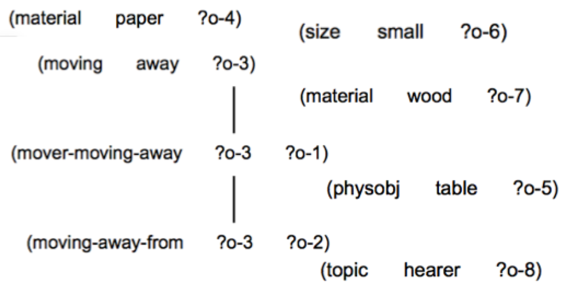


Figure 2: Disconnected network fragments resulting from lexicon lookup by the hearer for the utterance "wooden small moving-away table paper" (no syntax intended).

these variables. Given the world model in Figure 1, the following is a valid set of bindings for the network in Figure 2: {(?o-1 o-1) (?o-4 o-1) (?o-6 o-1) (?o-2 o-2) (?o-5 o-2) (?o-7 o-2) (?o-3 o-3)}. Note that some variables (such as ?o-1 and ?o-5) get bound to the same object. They are said to be *co-referential*.

The hearer can discover the bindings by matching the network that he derived from lexicon-lookup against his own world-model. But this approach, although in principle feasible and undoubtly used by human listeners, has a number of short-comings:

(a) The hearer needs to consider a rapidly exploding set of possible hypotheses $\mathcal{H}_n$. Concretely, H_n, the number of hypotheses, is equal to the number of partitions of the set D of words in an utterance of size n, where a partition of D is defined as a set of nonempty, pairwise disjoint subsets of D whose union is D. H_n is known as the Bell number and defined using the following equation (Bell, 1938):

$$H_{n+1} = \sum_{k=0}^{n} \binom{n}{k} H_k \qquad (1)$$

with $H_0 = H_1 = 1$. So H_n grows double exponentially with the number of words in the utterance. It means that as soon as an utterance is longer than a few words, it is not feasible anymore to rely exclusively on the world model.

(b) But even if the world model would give a possible set of bindings, there can still be remaining semantic ambiguity if there is more than one set of bindings compatible with the hearer's world model. This happens quite often in real dialog because the speaker is not distinctive enough in selecting the subnetwork of his world model that could uniquely identify the topic or assumes facts to be present in the hearer's world model which are not there.

(c) Interpretation through the world model is not possible in the case of displaced communication, where speaker and hearer do not share the same physical context and hence do not have a common world model (for example when speaking on the phone). And even if they share the same situation, the world-models of situated embodied agents are always different due to differences in perception, a different perspective on the scene, and a different focus of attention.

(d) And finally, even if bindings for all variables could be deduced by matching with the world model, the hearer might still not know which object is intended to be the topic.

Syntax can help because it can signal that two variables have to be bound to the same object, i.e. that they are co-referential. Indeed, if the speaker says "the small paper moves away from the wooden table", he signals that the facts introduced by "small" and "paper" as well as "wooden" and "table" pertain to the same object, i.e. that the variables introduced in the meanings of these words are co-referential. The hearer does this by grouping the words together and order them sequentially to express a noun-phrase pattern. The speaker also signals through the ordering of the constituents at the sentence level and the agreement for number and person between the subject and the verb that "the small paper" (the subject) is the mover and "the wooden table" (the direct object) is the object the paper moves away from. So syntax provides information to link network fragments into a globally coherent network, as in Figure 3. This network can then be matched against the world model to find the actual bindings, but, depending on how effective syntax has been, there will be almost no combinatorial search needed. But often some residual uncertainty remains and has to be disambiguated through the world-model. For example, in the sentence "Maria wants the pyramid on top of the block" it is not clear whether Maria wants the pyramid itself or wants to see it on top of the block.

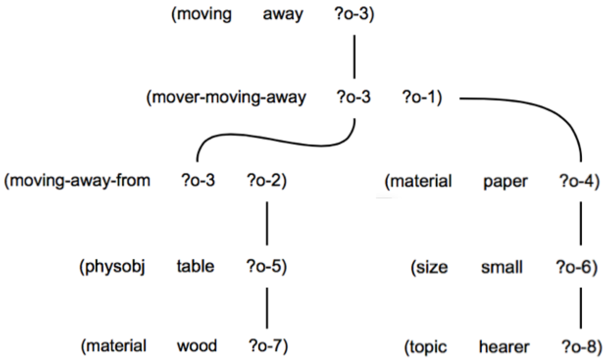


Figure 3: Co-referential links in the hearer network have been linked prior to matching with the world model and a topic has been added.

Reducing syntactic uncertainty

Now we turn to the issue of grammar. In principle, the Syntax Game is neutral with respect to which framework is adopted, as long as the grammar is bi-directional: it should support the transformation of meaning (in this case a semantic network) into an utterance for language production, and the reconstruction of the meaning of an utterance for language comprehension.

There is a broad consensus in linguistics that the bidirectional mapping of meaning to form proceeds through the intermediary of various linguistic units corresponding to words and phrases and syntactic and semantic categorizations of these units (see Figure 4). Examples of syntactic categorizations are parts of speech also known as lexical categories (Noun, Verb, etc.), agreement features (such as person, gender and number), temporal categories (tense, aspect, modality), syntactic functions (subject, direct-object, etc.), syntactic cases (nominative, accusative), etc. Examples of semantic categorizations include semantic categories (such as event-type, animacy), semantic roles (agents, patients) and case frames.

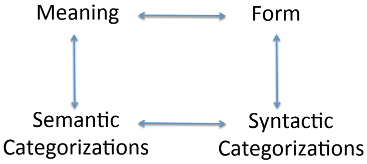


Figure 4: The bi-directional mapping between meaning and form goes through the intermediary of syntactic and semantic categorizations associated with linguistic units such as words and phrases.

There is also a consensus that the mappings of meaning, form, and unit categorizations are packaged in terms of constructions (Fillmore, 1988) and several computational formalisms have been developed recently to operationalize language processing in terms of such constructions. For the simulations reported later, we have used Fluid Construction Grammar (FCG) (Steels, 2011).

Thus a lexical construction creates a unit for a word and associates a word string with meaning and syntactic and semantic categorizations. For example, a lexical construction for the word "paper" associates the string "paper" with the meaning *(material paper ?obj)* and it categorizes the word from a syntactic point of view as a singular noun and from a semantic point of view as being material and having the referent *?obj*. In FCG notation, the lexical construction for "paper" is written as follows:

$$\left[\begin{array}{l} \textbf{?word} \\ \hline\hline \text{referent: ?obj} \\ \text{sem-cat: material} \\ \text{lex-cat: noun} \\ \text{number: singular} \end{array}\right] \rightarrow \left[\begin{array}{l} \textbf{?word} \\ \hline\hline \text{\# meaning:} \\ \quad \{\text{(material paper ?obj)}\} \\ \hline \text{\# form:} \\ \quad \{\text{(string ?word paper)}\} \end{array}\right]$$

On the right hand side are the conditions for activating this construction either in production (above the line) or in parsing (below the line). On the left hand side are the features that are to be added when the construction has become active.

A grammatical construction creates higher order units and determines their syntactic and semantic properties. For example, the sequential occurrence of an adjective and a noun introduces a new unit categorized as a noun phrase, and the meanings of the different constituents are related to each other by establishing co-referential relationships between their variables. Grammatical constructions also assign one of the constituents to be the head and this choice determines other properties (such as the referent) of the higher-order unit. A construction may also introduce new meaning. For example, the di-transitive construction, which underlies sentences such as "he bakes her a cake", adds the meaning that the indirect object ("her") is the recipient of a transaction caused by the subject ("he") and involving the direct object ("a cake") (Goldberg, 2006).

Here is an example of a (very simplified) grammatical construction in FCG-notation: It combines an adjective *?word-unit-1* and a noun *?word-unit-2* into a new hierarchical unit *?np-unit*:

$$\left[\begin{array}{l} \textbf{?np-unit} \\ \hline\hline \text{constituents:} \\ \quad \{\text{?word-unit-1, ?word-unit-2}\} \\ \text{sem-cat: material} \\ \text{syn-cat: noun-phrase} \\ \text{head: ?word-unit-2} \\ \text{referent: ?obj} \end{array}\right] \rightarrow$$

$$\left[\begin{array}{l} \textbf{?word-unit-1} \\ \hline\hline \text{sem-cat: size} \\ \text{referent: ?obj} \\ \hline \text{syn-cat: adjective} \end{array}\right] \leq \left[\begin{array}{l} \textbf{?word-unit-2} \\ \hline\hline \text{sem-cat: material} \\ \text{referent: ?obj} \\ \hline \text{syn-cat: noun} \end{array}\right]$$

The details of this formalism are not important here. The reader should just remember that the grammar operates through packages of associations between meaning, form, and unit categorizations and that the emergence of a grammar implies not only that agents come up with new grammatical constructions but also that they decide what hierarchical units are needed and what possible syntactic and semantic categories their grammar employs.

Language processing uses a basic data structure (often called the *transient structure*) to represent all units and features for one hypothesis on how to parse or produce an utterance. Language production starts from a transient structure which contains only the meaning of what needs to be expressed and then different constructions are applied until the transient structure contains enough information to articulate the utterance. Language comprehension starts from a transient structure that contains only information about what could be observed in the speech or written form and constructions are again applied until the meaning of the utterance can be extracted and interpreted by matching against the world model. In both cases search becomes unavoidable as soon as more than one construction gets triggered for the same transient structure. It is well known that this kind of search is also combinatorially explosive.

The world model can be used to reduce search. A transient structure, even if it has not yet processed all words, can often already be partly interpreted and some possible branches can then already be cut off because they are incompatible with the current world model. Human language users certainly use this approach, relying also partially on an ontology to weed out semantically incoherent combinations. But a hearer will encounter the same problems as discussed earlier for semantic interpretation. It would be much better if the grammar is more tight, i.e. uses additional grammatical features or provides additional syntactic marking, to avoid syntactic search. For example, the subject of a sentence is normally the first constituent of the sentence, but this can be made more explicit by requiring agreement between the subject and the verb for number and person.

A constructivist strategy for the Syntax Game

Strategies for building language cannot use most of the standard statistical machine learning techniques because there is no corpus to learn from and learning must be incremental: agents have to build up new grammar through successive situated language games. Instead, we use a constructivist strategy, where agents build hypotheses about what the communal language should or could look like, and adjust their hypotheses based on further interactions. The consecutive application of constructions expands the transient structure to go from meaning to form or vice versa. But occasions will arise when no construction can be applied. At such a point, a strategy for dealing with this impasse should become active (Figure 5). The strategy decomposes into 3 types of meta-operators: Semantic and syntactic meta-operators that try to repair the impasse based on the world model or on stretching existing constructions for new purposes, and learning meta-operators that become active at the end of processing and store what was learned

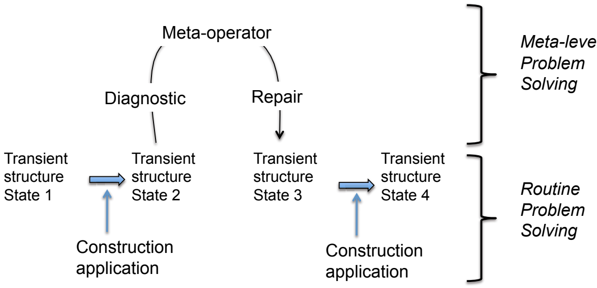

Meta-operator
Diagnostic
Repair
Meta-level Problem Solving
Transient structure State 1
Transient structure State 2
Transient structure State 3
Transient structure State 4
Construction application
Construction application
Routine Problem Solving

Figure 5: When agents do not have a construction to further expand a transient structure, they move to a meta-level, diagnosing the situation, possibly using the world-model or expansions of the grammar, and then continue routine processing based on constructions.

Semantic meta-operators

When no (partially) matching constructions can be found, it is possible to use the world model and combine units for words or phrases based on semantics, specifically:

+ **Build-or-extend-group**: If two words or word groups expressing unary predicates refer to the same object, they can be combined. For example, if there is a group for *wooden table* (based on an existing construction) and the utterance is *small wooden table*, the word-unit for *small* can be linked in with the group-unit for *wooden table*. The group-unit retains the same referent.

+ **Build-Hierarchy** When a relational word is encountered, i.e. a word which introduces a predicate with more than one argument, such as *moves-away-from*, and no constructions are available to handle this word, then the Build-Hierarchy meta-operator looks in the world-model to detect which object plays which roles and then combines the units for these objects into a new hierarchical unit.

The Build-Hierarchy meta-operator also decides which of the arguments is going to be the referent depending on the role of the arguments in the rest of the semantic network and determines on that basis the phrasal category of the hierarchical unit as well as its semantic category. For example, suppose the speaker must express an on-top-of relation with two arguments for the top and the bottom, then the head will be the unit introducing the top if the top is the object that links into the rest of the semantic network as in "the pyramid on top of the block". The hierarchical unit will be a noun-phrase (because the pyramid is already a noun-phrase) and its semantic category will be physical object.

Note that the Build-Hierarchy meta-operator leads to recursive syntax, because a noun-phrase (such as the block in "the pyramid on top of the block") is itself a constituent of a noun-phrase and this can go on at several levels as in "the pyramid on top of the block sitting on the table standing on the floor inside the room".

Syntactic meta-operators

When partially matching constructions can be found, it is possible to handle the impasse by either coercing words to fit into the partially matching construction (syntactic coercion) or to expand the applicability of the construction by making it more general (extension). More specifically,

+ **Coercion:** A construction is found that is semantically compatible but one word does not have the appropriate lexical category (as in the example of "googled" where a noun occurs in a context where a verb is expected). The Coercion meta-operator then adds the lexical category to the word-unit in the transient structure and the construction can apply.

+ **Extension:** A construction is found for which the syntactic constraints match but a required semantic category is missing. The Extension meta-operator then adds this semantic category to the construction so that it can apply.

Learning Operators

When an utterance could be successfully parsed or produced after one or more repairs, the learner activates learning operators to integrate the insights that were obtained into his construction inventory. The following learning operators have been implemented:

+ **Memoization**: This learning operator acts on the result of the semantic operators (build-or-extend-group and build-hierarchy). It builds a new construction, which should trigger when observing the relevant subunits and create a new superunit. There are two cases. (i) When the units are words expressing unary predicates, the head is the unit with the most referential power. (The hierarchy of referential power is provided by the ontology, e.g. physobj has more referential power than color.) The syntactic category of the superunit is equal to the phrasal variant of the lexical category of the head if it is a word, e.g. noun-phrase if the head is a noun, or the same phrasal category, if the head is already a phrase. And the lexical categories of the respective subunits are either randomly chosen from the existing categories, or, if there are no such categories associated already with the implicated word in the lexicon, a new lexical category is created and stored. The new categories are labeled *syn-cat-1*, *syn-cat-2*, etc. (ii) When the superunit is formed to handle a relational word, the head is the chosen referent, and the syntactic category of the superunit is the phrasal variant associated with the head (e.g. prepositional-phrase if the head is a preposition).

+ **Enact-Coercion**: This learning operator records the result of coercion by storing the assumed lexical category in the lexical construction of the relevant word.

+ **Enact-Extension**: This learning operator records the result of extension by expanding the set of semantic categories of the construction that was extended.

Alignment

The meta-operators and learning-operators are hypotheses made by the speaker and the hearer. Because neither of them has an absolute overview of the language and cannot inspect the internal states of the other agents, some hypotheses may be erroneous. The agents therefore need an additional mechanism to progressively discard wrong hypotheses based on further interactions. Theoretical research on the Naming Game has shown that it is best to let only the hearer adjust these scores De Vylder and Tuyls (2006) and to use a lateral inhibition learning rule, one of the strategies commonly used for the Naming Game Steels (1998). Knowing which constructions c_i need an increased score is easy: they are the constructions that were used on the path towards the final transient structure. We use the following update rule: $\sigma_{c_i} \leftarrow \sigma_{c_i}(1-\gamma)+\gamma$, with $\gamma = 0.1$ a constant.

Competing constructions c_j need to be decreased using the following update rule: $\sigma_{c_j} \leftarrow \sigma_{c_j}(1-\gamma)$. How can the agent determine competing constructions? First of all, they include all constructions that started off a wrong branch in the search space during comprehension, i.e. a branch which was not on the path towards the final solution. When such constructions are discarded, this will minimize the syntactic uncertainty S_t. Second, the listener can produce himself the utterance based on the meaning deduced from the comprehension process and then find all constructions that would start off a wrong branch in producing, i.e. a branch that would not lead to the utterance produced by the speaker. Their scores need to be decreased as well.

Results

Because the Syntax Game focuses on grammar, we start simulations from a pre-defined set of lexical constructions associating word strings with meanings. Semantic categories are directly derived from the attributes of the meaning. The agents must autonomously introduce and assign lexical categories to the words and introduce new hierarchical units, categories for these units and grammatical constructions that build them. Each agent has to do this independently but a shared common grammar has to emerge through self-organization.

The first simulation experiment tests whether an agent is able to learn an existing grammar in one-on-one interactions. The tutor is initialized with a lexicon of 40 lexical constructions and a grammar with 30 grammatical constructions. The tutor grammar includes adverbs, adjectives, nouns, verbs, prepositions, pronouns and relative pronouns as well as noun phrases of different levels of complexity, verb phrases, main clauses and relative clauses. The constructions produce utterances in a reduced English, without articles and without grammatical agreement. Each sentence describes a particular topic (object or event) in a scene. Some example utterances are "Paul sees (the) red carpet (that) Emilia wants", "big red table on small stone" or "Paul believes (that) Emilia wants (the) carpet on (the) big wooden table". The learner is initialized with the same lexicon and endowed with the various operators described above, but without any grammatical constructions or lexical and phrasal categories. Each experiment is carried out for 5 different tutor-learner pairs, using random choices from a set 20 situations, so that results are comparable.

Words have potentially more than one lexical category (as in human languages), e.g. "paper" can be a noun as well as an adjective (as in a paper towel), although only one lexical category can fit with a grammatical construction. On the other hand a grammatical construction can accept more than one semantic category for a particular unit, which then makes the construction more general from a semantic point of view.

Figure 6 shows the result of a tutor-learner experiment. It shows 500 consecutive language games. The running average of four different measures is shown for 2 different experiments:

(i) The *semantic uncertainty*, which measures the number of times the world model of the situation is used to generate or block hypotheses, divided by the number of variables introduced by the lexicon for the utterance. More precisely, when grammar or syntactic meta-operators have been used, it measures the number of hypotheses that were not plausible according to the world model and when semantic meta-operators have been used, it measures the number of times the situation was used to generate possible hypotheses.
(ii) The *syntactic uncertainty*, which measures the number of extra hypotheses that grammatical constructions generated during processing. This is the number of splits in the search space that were due to multiple results of applying a grammatical construction, divided by the number of words in the utterance.
(iii) The *communicative success*, which measures whether the hearer was able to identify the topic without speaker feedback.
(iv) The *alignment* which measures whether the hearer would express the same meaning using the same utterance as the speaker.
(v) The *grammatical constructions* measure, which measures how fast the grammar is acquired by measuring the percentage of constructions learned of the final grammar.
Thanks to the grammar we see that both types of uncertainty get drastically reduced, which proves that the grammar achieves its desired purpose of minimizing uncertainty.

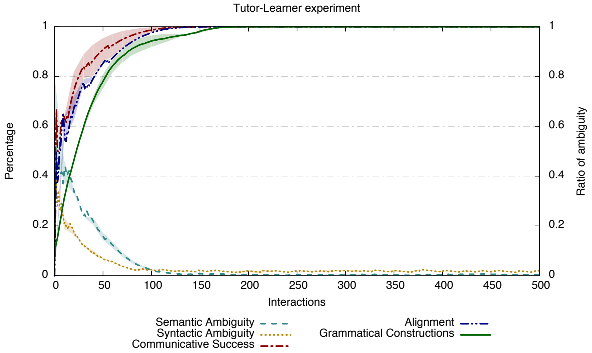


Figure 6: Tutor-learner experiment. The learner rapidly (after 150 interactions) acquires the grammar of the tutor, and a total alignment and communicative success with the tutor. On the other side, semantic and syntactic uncertainties of the learner decrease drastically as long as grammar is learned.

Figure 7 shows the results of an experiment in which a population of 5 agents develops a new grammar from scratch. The agents start with a shared lexicon but no syntactic categories and no grammatical constructions. They build very quickly a common grammar that provides high communicative success, total alignment as well as a reduction of syntactic and semantic uncertainty.

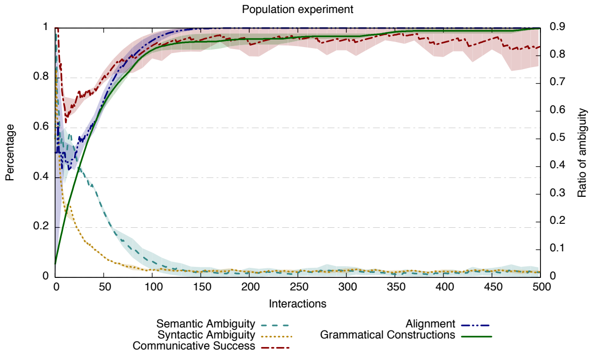


Figure 7: Grammar emergence experiment. The same measures as in 6 are shown.

What about the complexity of the grammar? We observe that the agents construct noun-phrase-like constructions with categories reminiscent of adjectives and nouns. They also build relational constructions with lexical categories similar to prepositions and verbs as well as relative clauses such as "(the) red carpet (where) emilia wants (the) table (to be) on." There are many examples of recursive structures, particularly embedded clauses and embedded noun-phrases.

It is not straightforward to compare the grammars of two agents directly because agents have different syntactic categories. To visualise nevertheless the similarity between syntactic categories we have used the multidimensional scaling method. In order to apply it, we have defined a vector space to describe every syntactic categories as a vector in the following way: $Cat_i \in \{0, 1\}^n$ where n equals the number of words in the lexicon and position j is 1 when word w_j can be used as category Cat_i, and 0 when it cannot. Finally we have applied the method to these corresponding sets of vectors by using the Euclidean distance. Figure 8 shows results from the Tutor-Learner experiment where we see that the learner is able to form categories similar to the syntactic categories that the tutor uses.

Conclusion

This paper introduced the Syntax Game, a minimal language game for exploring how grammar can arise in a population of agents. The game is a variant of the Naming Game but introduces more complex semantics. Grammar is needed to avoid semantic and syntactic uncertainty which lead, respectively, to a combinatorial explosion in semantic interpretation and in parsing. Hence a language cannot scale up to utterances beyond a few words without minimizing these sources of utterance in language processing. We discussed an example of a language strategy, which allows agents to self-organize a phrase structure grammar that uses sequen-

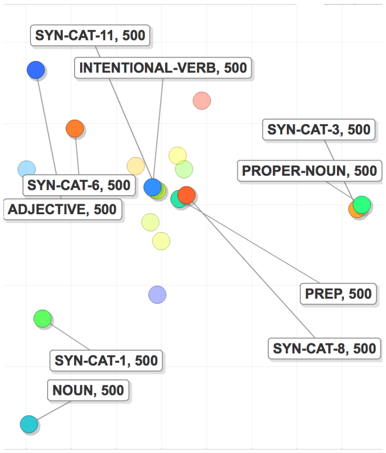


Figure 8: MDS plot, useful for showing how syntactic categories of two agents become similar.

tial ordering and grouping of words into phrases in order to avoid semantic and syntactic uncertainty. This strategy uses a set of meta-operators and alignment based on a lateral inhibition learning rule. Experimental results show that the strategy achieves the desired results.

Acknowledgments

The research reported here has been funded by an ICREA Research fellowship to LS and by the EU FET-Open Insight Project and a Marie Curie Integration Grant EVOLAN.

References

Bell, E. (1938). The iterated exponential integers. *The Annals of Mathematics*, 39.

De Vylder, B. and Tuyls, K. (2006). How to reach linguistic consensus: A proof of convergence for the naming game. *Journal of Theoretical Biology*, 242(4):818–831.

Fillmore, C. J. (1988). The mechanisms of “Construction Grammar”. In *Proceedings of the Fourteenth Annual Meeting of the Berkeley Linguistics Society*, pages 35–55, Berkeley CA. Berkeley Linguistics Society.

Goldberg, A. E. (2006). *Constructions at Work, The Nature of Generalization in Language*. Oxford University Press, Great Clarendon Street, Oxford.

Loreto, V., Baronchelli, A., Mukherjee, A., Puglisi, A., and Tria, F. (2011). Statistical physics of language dynamics. *JOURNAL OF STATISTICAL MECHANICS: THEORY AND EXPERIMENT*, P04006.

Spranger, M., Loetzsch, M., and Steels, L. (2012). A Perceptual System for Language Game Experiments. In Steels, L. and Hild, M., editors, *Language Grounding in Robots*, pages 89–110. Springer.

Steels, L. (1995). A self-organizing spatial vocabulary. *Artificial Life Journal*, 2(3):319–332.

Steels, L. (1998). The origins of ontologies and communication conventions in multi-agent systems. *Journal of Agents and Multi-Agent Systems*, 1(2):169–194.

Steels, L., editor (2011). *Design Patterns in Fluid Construction Grammar*. John Benjamins.

Steels, L. and Hild, M., editors (2012). *Language Grounding in Robots*. Springer, New York.

How Much Should You Select for Evolvability?

Andrew M. Webb[1], Julia Handl[2] and Joshua Knowles[1]

[1]School of Computer Science, The University of Manchester, UK
[2]Decision and Cognitive Sciences Group, MBS, The University of Manchester, UK
andrew.webb@manchester.ac.uk

Abstract

We consider whether selection for evolvability leads to greater adaptive progress than selection for adaptedness alone. Our treatment bears on longstanding discussions of selection for evolvability in the literature, which have been largely limited to conceptual and qualitative arguments to date. We study a simple mathematical model of a population of individuals whose adaptedness and evolvability (here modelled as the standard deviation of mutations affecting adaptedness) are both under selective forces. In the special case of a population of size two, we show that the optimal amount of selection for evolvability depends on the ratio between the initial evolvability and the amount that evolvability can increase in the time given. Our result shows that to maximize the amount of adaptation it never pays off to select for evolvability more than to select for adaptedness itself. We have not answered the question of to what degree evolvability is selected for in nature, however we have made a small step in quantitative modelling of the evolution of evolvability and proved the existence of conditions under which selection for evolvability has a demonstrably positive effect.

Introduction

The definition of evolvability has been hard to pin down (Kirschner and Gerhart, 1998). One popular definition is that it is a property of an individual, or an aggregate property of a lineage, that increases the expected rate of adaptation of the lineage. For example, for Hansen (2006) evolvability is "the ability of the genetic system to produce and maintain potentially adaptive genetic variants." All measures of this type of evolvability depend in some way on the probability distribution of fitness effects of mutations (Altenberg, 1995). Proposed measures include the likelihood of a mutation being beneficial (Smith et al., 2002) and the variance of the fitness of offspring prior to selection (Gallagher, 2009).

The evolution of evolvability has become a popular research topic for biologists (Dawkins, 2003; Pigliucci, 2008), researchers of evolutionary computation (Altenberg, 1994; Reisinger and Miikkulainen, 2006), and in the artificial life community (McMullin, 2012; Webb and Knowles, 2014). The questions being asked fall into two broad categories:

1. Has evolvability increased through evolution in life on Earth? If so, by what mechanisms?
2. How can we encourage the evolution of evolvability in evolutionary algorithms and artificial life simulations?

In this paper, we ask *to what extent* we should select for evolvability in simulated evolution in order to maximize the degree of adaptation overall. To our knowledge, this has not been done before. We have in mind a scenario in which adaptedness and evolvability are simultaneously under selective forces, and where the two are inextricably linked such that there is a trade-off; an individual can't inherit its adaptedness from one parent and its evolvability from another. This would be the case, for example, if an individual's evolvability were determined by the way in which its traits are encoded in the genetic material that undergoes variation. If one individual is well adapted, but another, less well adapted, individual is more evolvable due to having a more suitable encoding, in general it is not straightforward to re-encode the more adapted individual using the better encoding.

An answer to the question of to what extent we should select for evolvability will primarily be useful in evolutionary computation and artificial life simulations, where often the goal is to maximize the degree of adaptation. It may be less useful in answering questions about natural evolution, where there is no such forward planning, though it still might provide a useful piece in a larger puzzle.

Our contributions are as follows.

- We introduce a simple model of a population in which adaptedness and evolvability are simultaneously under selective forces, and in which we control the relative importance of adaptedness and evolvability during selection.
- For the special case that the population is of size two, we answer exactly to what extent evolvability should be selected for (to the exclusion of selecting for adaptedness) to maximize the total amount of adaptation.
- We discuss the difficulties of using the same method to analyze a larger population.

- We list the ways in which the model might be extended in order to better reflect realistic scenarios.

The Model

We have a population of N individuals, each with two traits A and B. The A value of an individual represents adaptedness to some environment, and as such it is a value to be maximized. The B value of an individual represents the "evolvability with respect to trait A"; it is the key factor in determining the rate of increase of A in the course of evolution. Here, that means that the Bs determine the standard deviations of mutations affecting the As.

In each generation, we rank the population by the value $\gamma A + (1 - \gamma)B$, where the As and Bs have first been normalized by dividing by the standard deviations of those traits in the population. The weighting parameter γ is under our control, and takes values in the range $[0, 1]$. This parameter represents a trade-off between selecting for adaptedness and evolvability. The top proportion p become the parents of the next generation; we select with replacement from the set of parents to form the next generation.

The reason we include the normalization step is that, without it, as the variance of one of the traits in the population becomes large, evolution stops acting on the other trait, regardless of the value of γ. This is illustrated in Figs. 1 and 2. These show, for a large population with normally distributed A and B traits, the expected increase in the population mean values of A and B if we select the top half by value $A + B$, where the standard deviation of the A values is 1 and the standard deviation of the B values is parameterized by β. Without normalization, as Fig. 1 shows, the expected increase of each trait is a nonlinear function of both of the trait standard deviations. As the standard deviation of the Bs, β, tends to infinity, the expected increase in trait A due to selection tends to zero. With normalization, as Fig. 2 shows, the expected increase of each trait is proportional to the standard deviation of just that trait.

After the selection step, we mutate the A and B values as follows. We add Gaussian noise to each individual's B value with a constant standard deviation, β. We add Gaussian noise to each individual's A value with standard deviation αB (i.e., a constant times that individual's B value). Since standard deviations must be positive, we prevent the Bs from taking negative values; when a mutation makes a B value negative, we set it to a small positive value ϵ.

Algorithm 1 shows the process of evolution in this model. Table 1 lists the parameters and their roles.

Our question, stated in terms of the model, is as follows. What value of the parameter γ maximizes, at some particular future time t_{end}, the expected mean value of A? We answer this question exactly in the special case that the population size $N = 2$ and the proportion selected as the parents of the next generation $p = 1/2$.

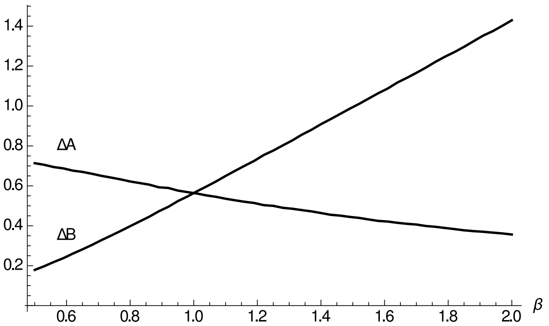


Figure 1: Without the normalization step, when we select for $A + B$, the expected increase in the mean value of each trait (ΔA and ΔB) is a nonlinear function of the standard deviations of both traits. As the standard deviation of the Bs, β, tends to infinity, trait A stops being selected for.

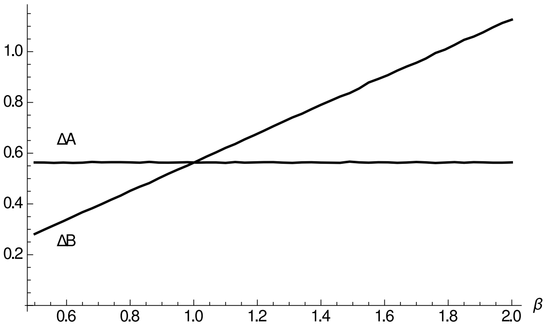


Figure 2: With the normalization step, when we select for $A + B$, the expected increase in the mean value of each trait (ΔA and ΔB) is a linear function of the standard deviation of just that trait.

Derivation of the Result

We restrict ourselves to the special case that the population size $N = 2$ and the proportion kept during selection $p = 1/2$. We can restate the problem so that we only have to keep track of one A value and one B value in each generation; let $A(t), B(t)$ be the A and B values of the *parent* for generation t, with $A(0) = A_0$, $B(0) = B_0$.

In each generation, we duplicate the parent, mutate both copies, and then, after normalizing by dividing the As and Bs by their population standard deviations, we select as the parent for the next generation the individual with the maximum value of $\gamma A + (1 - \gamma)B$. Since the parents are identical before mutation, we are essentially selecting between mutation events. The normalization step means that, as far as the selection operator is concerned, all mutations have a standard deviation of 1.

We can achieve the same result by drawing four 'normalized' mutations from the standard normal distribution $\mathcal{N}(0, 1)$. M_{A_1} and M_{A_2} are the normalized mutations af-

Algorithm 1 Our model of simultaneous evolution of adaptedness and evolvability.

```
Initialize vector $A(t=0)$ with $N$ elements of value $A_0$
Initialize vector $B(t=0)$ with $N$ elements of value $B_0$
for each $t \leftarrow 0..t_{end}$ do
    $A' \leftarrow A(t)/\text{std\_dev}(A(t))$
    $B' \leftarrow B(t)/\text{std\_dev}(A(t))$
    Sort $A(t)$, $B(t)$ by the corresponding value $\gamma A' + (1-\gamma)B'$
    for each $n \leftarrow 1..N$ do
        $i \leftarrow$ random variate drawn from the discrete uniform distribution $[1, pN]$
        $M_A \sim \mathcal{N}(0, \alpha^2 B(t)[i]^2)$
        $M_B \sim \mathcal{N}(0, \beta^2)$
        $A(t+1)_n \leftarrow A(t)_i + M_A$
        $B(t+1)_n \leftarrow B(t)_i + M_B$
        if $B(t+1)_n < 0$ then
            $B(t+1)_n \leftarrow \epsilon$
```

N	—	The population size. Here we set $N = 2$.
p	—	The proportion of individuals chosen by truncation selection as parents of the next generation. Here we set $p = 1/2$.
A_0	—	The initial value of the trait A in the population.
B_0	—	The initial value of the trait B in the population. Non-negative.
α	—	A parameter adjusting the standard deviation of A mutations. Non-negative.
β	—	A parameter adjusting the standard deviation of B mutations. Non-negative.
t_{end}	—	The time at which we want to maximize the expected mean value of A, with respect to the parameter γ. Positive integer.
γ	—	A parameter under our control representing a trade-off between selection for the traits A and B. In the range $[0, 1]$.

Table 1: Parameters of the model.

fecting the As, and M_{B_1} and M_{B_2} are the normalized mutations affecting the Bs. We will then select the pair of mutations with the largest value of $\gamma M_A + (1-\gamma)M_B$, and multiply each by the desired mutational standard deviation, undoing the normalization step. We then apply these mutations to the parent to get the A and B values of the parent of the next generation.

Let $\langle M_A^+, M_B^+\rangle$ be the $\langle M_A, M_B\rangle$ pair with the maximum value of $\gamma M_A + (1-\gamma)M_B$. That is,

$$\langle M_A^+, M_B^+\rangle = \langle M_{A_m}, M_{B_m}\rangle, \text{ where} \tag{1}$$
$$m = \operatorname*{argmax}_{i\in\{1,2\}} \big(\gamma M_{A_i} + (1-\gamma)M_{B_i}\big).$$

The two components in the sum in (1) are distributed as

$$\gamma M_{A_{1,2}} \sim \mathcal{N}(0, \gamma^2) \tag{2}$$
$$(1-\gamma)M_{B_{1,2}} \sim \mathcal{N}(0, (1-\gamma)^2). \tag{3}$$

Using the results (21), (22) from the appendix, we obtain the expected values of these components in the pair with the maximum sum, which are

$$E[\gamma M_A^+] = \frac{\gamma^2}{\sqrt{\gamma^2 + (1-\gamma)^2}\sqrt{\pi}} \tag{4}$$
$$E[(1-\gamma)M_B^+] = \frac{(1-\gamma)^2}{\sqrt{\gamma^2 + (1-\gamma)^2}\sqrt{\pi}}. \tag{5}$$

From one generation to the next, A will increase by $\alpha B(t)M_A^+$, and B will increase by βM_B^+. These are the 'unnormalized' mutations, which have the expected values (taking the expectation over the possible values of M_A^+, M_B^+)

$$E[\alpha B(t)M_A^+] = \frac{\gamma}{\sqrt{\gamma^2 + (1-\gamma)^2}\sqrt{\pi}}\alpha B(t) \tag{6}$$
$$E[\beta M_B^+] = \frac{1-\gamma}{\sqrt{\gamma^2 + (1-\gamma)^2}\sqrt{\pi}}\beta. \tag{7}$$

The nonlinear trade-off between the rates of increase of A and B, determined by the parameter γ, is shown in Fig. 3, and is given by

$$f(\gamma) = \frac{\gamma}{\sqrt{\gamma^2 + (1-\gamma)^2}}. \tag{8}$$

Since $\alpha B(t)M_A^+$ and βM_B^+ represent the change in A and B from generation t to $t+1$, we have the recurrence relations

$$A(t+1) = A(t) + \alpha B(t)M_A^+(t) \tag{9}$$
$$B(t+1) = B(t) + M_B^+(t), \tag{10}$$

and the expected value of $A(t)$ and $B(t)$ (now taking the expectation over the mutation events in *every* generation) satisfy the recurrence relations

$$E[A(t+1)] = E[A(t)] + f(\gamma)\frac{\alpha E[B(t)]}{\sqrt{\pi}} \tag{11}$$
$$E[B(t+1)] = E[B(t)] + f(1-\gamma)\frac{\beta}{\sqrt{\pi}}. \tag{12}$$

Solving the second recurrence relation gives the expected value of B at time t, which is

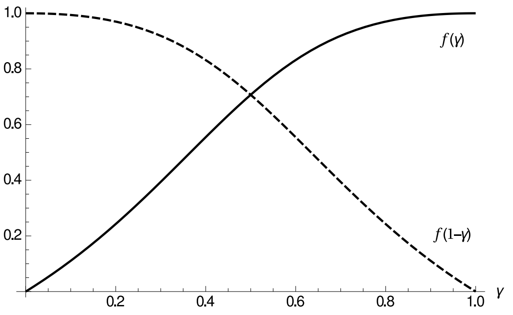


Figure 3: The trade-off between selecting for traits A and B in a population of size 2. The function $f(\gamma)$ gives the expected increase in trait A due to selection, in units of the standard deviation of A mutations. The function $f(1-\gamma)$ plays the same role for the expected increase in trait B.

$$E[B(t)] = B_0 + f(1-\gamma)\frac{\beta t}{\sqrt{\pi}}, \tag{13}$$

and solving the first recurrence relation gives the expected value of A at time t, which is

$$\begin{aligned} E[A(t)] = A_0 + \frac{\alpha t}{4\pi}\Bigg(& \frac{4\sqrt{\pi}B_0\gamma}{\sqrt{\gamma^2+(1-\gamma)^2}} \\ & - \beta(t-1)\left(1+\frac{1}{\gamma^2+(1-\gamma)^2}\right)\Bigg), \end{aligned} \tag{14}$$

To find a value of γ that maximizes the expected value of A at time t_{end}, we set the derivative of the above expression with respect to γ equal to zero. The solution γ^* in the range $[0,1]$ that satisfies this equation is

$$\begin{aligned} \gamma^* &= \frac{3}{4}+\frac{1}{4}\big(z-\sqrt{z+3}\sqrt{z-1}\big), \text{ where} \\ z &= \sqrt{1+\frac{2}{\pi}\left(\frac{\beta(t_{end}-1)}{B_0}\right)^2}. \end{aligned} \tag{15}$$

The optimal value of the trade-off parameter, γ^*, depends only on B_0, β, and t_{end}, and is an increasing function of $B_0/(\beta(t_{end}-1))$. Figure 4 shows γ^* as a function of $B_0/(\beta(t_{end}-1))$, while Fig. 5 shows γ^* as a function of the reciprocal $\beta(t_{end}-1)/B_0$. Both are shown so that both asymptotes are clear.

That γ^* depends on this quantity makes sense; it is the ratio between the initial evolvability B_0 and βt_{end}, which is related to the amount that evolvability can increase in the course of evolution in the time given. If the initial evolvability is large compared to the capacity to increase evolvability, then it pays off to focus more on increasing the trait A. As we look further to the future and t_{end} becomes large, the initial evolvability value has less of an effect and γ^* tends towards 1/2. The optimal value γ^* is never less than $1/2$; it never pays off to select more for evolvability than for adaptedness.

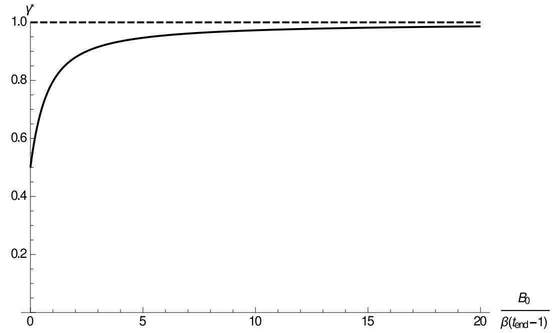


Figure 4: The optimal value of the trade-off parameter γ as an increasing function of $B_0/(\beta(t_{end}-1))$. As B_0 becomes large, the optimal value asymptotically approaches 1.

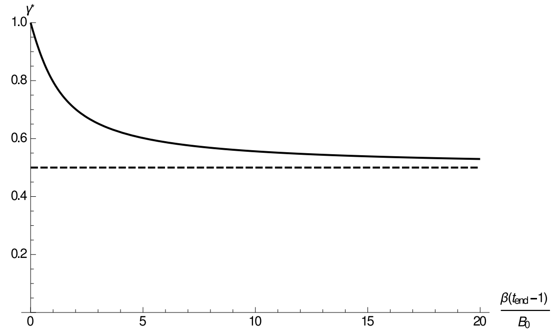


Figure 5: The optimal value of the trade-off parameter γ as a decreasing function of $\beta(t_{end}-1)/B_0$. As β or t_{end} become large, the optimal value asymptotically approaches $1/2$.

Figure 6 shows the optimal value of γ found by numerical methods for a range of values of B_0, β, and t_{end}. For each setting of the parameters, we plot the value of γ with the highest mean value of A at time t_{end} measured over one hundred thousand trials (the low population size make the outcome noisy). The numerical results closely agree with the answer obtained here, verifying the result[1].

Figure 7 shows, for a particular setting of the parameters, the expected value of A over time for three strategies; setting $\gamma = 1$ (so that only A is selected for), setting $\gamma = 1/2$ (so that we select equally for A and B), and setting $\gamma = \gamma^*$ (the optimal value). It can be seen that the γ^* strategy dominates. Figure 8 shows the same with a different setting of the parameter β. Note that for the γ^* strategy, the plots

[1] There is a small discrepancy, because our result does not account for the fact that, after a mutation, we set negative B values to small positive values. This manifests when B_0 is small enough that B is small compared with β in the initial generations.

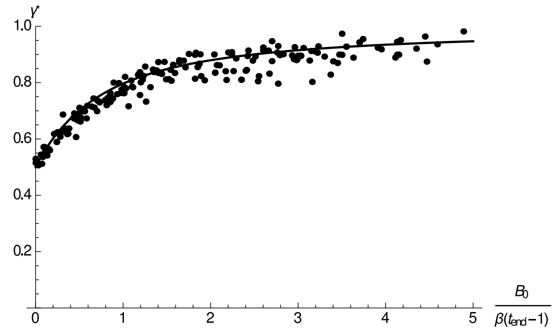


Figure 6: A comparison between the optimal value of γ obtained by numerical methods and the exact result.

do not show A over time for a particular value of γ; for each time t, the plot shows the expected value of A when using the (constant) value of γ that maximizes $A(t)$.

Larger Population Sizes

With a larger population size, the model is harder to analyze for two reasons. The first is that the trait variances in the population depend on mutations accumulated over multiple generations; because more than one parent is selected in each generation there will be residual variation from the previous generation. Moreover, this residual (post-selection) variation will not be normally distributed. The result is that the traits A and B will no longer be normally distributed, but will be skewed by an amount depending on the tradeoff parameter γ and the proportion kept during selection p. The trait distributions will change over time, approaching an equilibrium shape.

The second problem is that, because more than one parent is selected in each generation, correlation builds up between the A and B values in the population; individuals selected for having high A values are likely to have inherited large B values. The result of this correlation is that there is indirect selection for trait B when selecting for trait A.

Figure 9 shows (with $N = 100, p = 1/2$), as functions of γ, the measured mean increase in traits A and B during selection in generation 10, in units of the mutational standard deviation of A and B, respectively. Figure 10 shows the same in generation 50. The asymmetry is due to indirect selection for trait B, and the mean increase of each trait due to selection changes over time because both the trait distributions and the correlation between the traits are changing over time. Compare these with the stationary (in time) and symmetric functions giving the expected per-generation increase of traits A and B for a population of size two, shown in Fig. 3. Without an exact expression for the expected increase of traits A and B due to selection in a larger population, we cannot deduce the optimal value of the trade-off parameter γ.

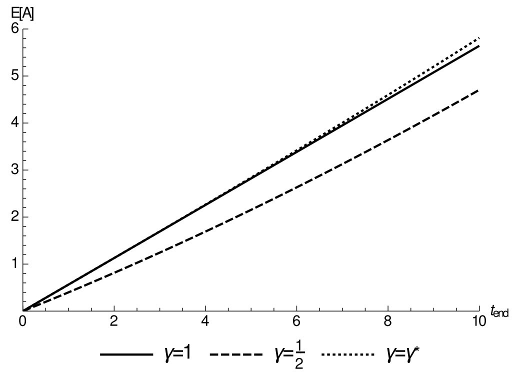


Figure 7: The expected value of A over time for three strategies for setting γ. $A_0 = 0, \alpha = 1, B_0 = 1, \beta = 0.1$.

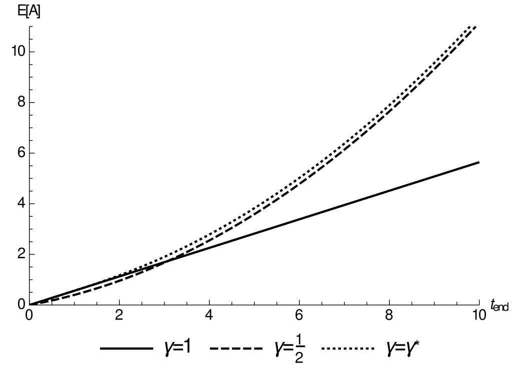


Figure 8: The expected value of A over time for three strategies for setting γ. $A_0 = 0, \alpha = 1, B_0 = 1, \beta = 1$.

Figure 11 shows the optimal value of γ found by numerical methods, with a population size $N = 100$ and $p = 1/2$. The exact result from the population size $N = 2$ case is shown for comparison.

Related Work

Here we review research related to selection for evolvability from the research fields of evolutionary biology, evolutionary computation, and artificial life. There has been much debate about how to define evolvability, and about whether it is a property of individuals or populations. In our work we have followed Conrad (1972), Altenberg (1994), Kirschner and Gerhart (1998), and others in defining evolvability to be a property of individuals, related to their amenability to adaptive evolution, or capacity to produce potentially more well-adapted offspring.

Even amongst those who agree with this definition, there is no consensus about exactly how to measure evolvability. Proposed measures include the likelihood of a beneficial mutation, the expected fitness of offspring after selection, and the expected variance in fitness of offspring prior to selection. Gallagher (2009) gives a summary of these and

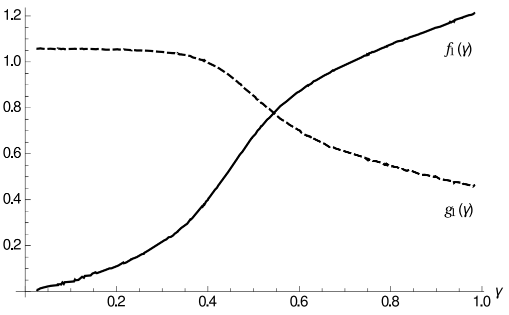


Figure 9: The function $f_1(\gamma)$ shows the measured mean increase in trait A in generation 10 in units of the standard deviation of A mutations. The function $g_1(\gamma)$ plays the same role for trait B. The functions are asymmetric; there is indirect selection for B when selecting for A. $N = 100$.

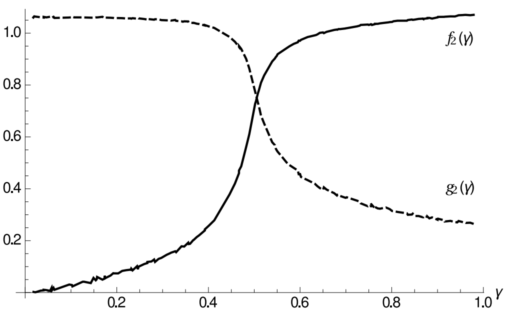


Figure 10: The function $f_2(\gamma)$ shows the measured mean increase in trait A in generation 50 in units of the standard deviation of A mutations. Function $g_2(\gamma)$ plays the same role for trait B. The functions are not the same as those for generation 10, and the functions are asymmetric. Compare with Fig. 3. $N = 100$.

other measures. It is this last measure, the pre-selection variance of offspring, which is closest to ours; in our model, the evolvability of an individual is the standard deviation of mutations affecting the individual's offspring.

In biology, questions have been asked about whether evolvability has evolved in life on Earth, and if it has, whether it evolved by natural selection or by some additional mechanisms (Pigliucci, 2008). The first question is motivated by the fact that evolvability confers a future, rather than present, advantage, and it's not obvious that evolution has or is able to select for evolvability even if doing so would be beneficial in the long term (Kirschner and Gerhart, 1998). Amongst those who believe evolvability has evolved, proposed mechanisms include indirect selection due to correlation between fitness and evolvability (Altenberg, 1994), direct selection for evolvability as an adaptation or as a byproduct of selection for environmental robustness (Hansen, 2006; Visser et al., 2003), higher level

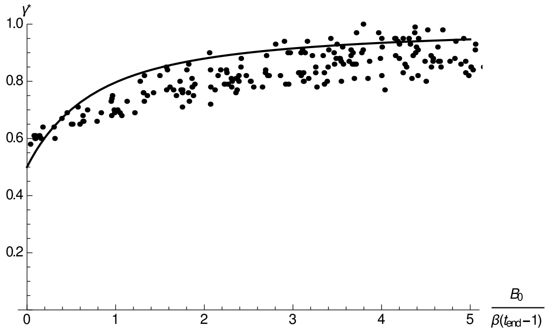


Figure 11: A comparison between the optimal value of γ obtained by numerical methods (with population size $N = 100$) and the exact result (with $N = 2$).

selection between clades and groups (Alberch, 1991), and a process of repeated extinctions and radiations (Dawkins, 2003).

In recent experiments in artificial life, the underlying encodings of self-replicators (i.e., the way in which the replicators are encoded in their heritable genetic information) has been allowed to evolve. The hope is that more evolvable encodings (with respect to the environment) will emerge. Many of these simulations were implemented in Avida and its variants. Avida is an artificial life platform in which assembly-like computer programs self replicate (Ofria and Wilke, 2004).

Baugh and McMullin (2013) and Hasegawa and McMullin (2013) have, respectively, designed replicators for the Tierra and Avida self-replication platforms in which part of the self-replicating program is interpreted as genetic information, and another part is interpreted as a decoding mechanism, that decodes the genetic information. By allowing both to evolve, the way that the replicators are encoded in the genetic portion can change over time.

Egri-Nagy and Nehaniv (2003) have implemented a variant of Avida in which each replicating program has its own, different, instruction set, which itself can evolve. The goal is similar; the way that a replicator's behaviour is encoded can change over time, and more evolvable encodings might be discovered.

Webb and Knowles (2014) aimed to study the differing capacities to evolve evolvability between 'non-self-encoding' and 'self-encoding' replicators. In both cases, each replicator implement a decoder that interprets its genetic information, and the decoder itself can evolve over time. In 'self-encoders', the decoder determines the way in which it *itself* is encoded in the genetic information. The authors concluded that there may have been insufficient selection for evolvability in their simulations to distinguish between the two types of replicator.

In the evolutionary computation field, there is a general concern with evolvability in the sense of mechanisms that might improve the capacity for adaptive evolution. Placing the degree of mutational variation under evolutionary control has been studied by Eiben et al. (1999), mostly from an empirical perspective, though important theoretical work in this area has also been done (Rudolph, 2001).

Reisinger and Miikkulainen (2006) list some ways in which evolvability has been allowed to evolve in evolutionary algorithms. For example, Ebner et al. (2002) study the evolution of evolvability in neutral networks, in which neighbouring genotypes can encode the same phenotype. Neutral mutations, whose relevance to evolvability was first outlined by Maynard Smith (1970), are those that leave the phenotype unchanged, while possibly changing the phenotypic neighbourhood. Reisinger and Miikkulainen give as other examples evolutionary algorithms with indirect encodings (Stanley and Miikkulainen, 2003) and Estimation-of-Distribution algorithms (Pelikan et al., 2002).

Altenberg (1994) shows that in genetic programming, evolvability can evolve by the implicit selection of blocks of code for what he calls their ‘constructional selection’, or their ability to improve programs in the population when inserted into them.

Conclusion and Future Work

We have introduced a model of a population simultaneously evolving an adaptive trait A and a trait B that is the “evolvability with respect to A”, with the aim of answering to what extent we should select for evolvability in order to maximize the total amount of adaptation over a given time period.

We have answered this question exactly in the special case that the population size is two, with one individual selected as the parent of the next generation. We find that the optimal weighting parameter γ is never less than $1/2$ and is an increasing function of $B_0/(\beta(t_{end}-1))$, asymptotically approaching 1, where B_0 is the initial evolvability, β is the standard deviation of mutations affecting B, and t_{end} is the time at which we want to maximize A with respect to γ.

The model is straightforward to analyze with a population size of two, because the two individuals only ever differ in the mutations that happen within the current generation, and the mutations are independent and normally distributed; there are no residual differences between individuals from earlier mutations and there is no correlation between the A and B values within the population. As a result, the expected increase in A due to selection is a function of γ times the standard deviation of A mutations, and the expected increase in B is the same function of $1-\gamma$ times the standard deviation of B mutations (see Fig. 3). For larger populations, the trait distributions are not normal, change over time, and become correlated, making analysis more difficult.

The applicability of our result is limited by the assumptions of the model, which are as follows.

- “Evolvability” is determined wholly by the B value, and is independent of time, environmental variables, and where each individual is on the A-landscape.
- The As and the Bs can increase without limit.
- We have perfect knowledge of the Bs. In practice, we would likely have to rely on estimates of the evolvability of an individual or lineage, derived from observations of the effects of past mutations.

In future work we aim to extend the model to overcome one or more of these limitations.

Appendix

If we have two random variables A and B both distributed as $\mathcal{N}(0,\sigma^2)$, then the maximum of A and B has the expected value

$$\begin{aligned} E[max(A,B)] &= \int_{-\infty}^{\infty} a\phi(a) \int_{-\infty}^{a} \phi(b)\,\mathrm{d}b\,\mathrm{d}a \\ &+ \int_{-\infty}^{\infty} b\phi(b) \int_{-\infty}^{b} \phi(a)\,\mathrm{d}a\,\mathrm{d}b \\ &= 2\int_{-\infty}^{\infty} a\phi(a) \int_{-\infty}^{a} \phi(b)\,\mathrm{d}b\,\mathrm{d}a, \end{aligned} \tag{16}$$

where $\phi(x)$ is the pdf of the distribution. This can be understood as follows. We integrate over the possible values of A, multiplying the probability of getting that value by the probability that the B value is less than it (i.e., the probability that the A is the maximum of the pair). For each possibility we multiply by the value of A to get the expected value. We then do the same thing for the case where the B value is the greater of the pair. Because A and B have the same distributions, these integrals are equal, so we evaluate it once and double the result. Evaluating the integral gives the result

$$E[max(A,B)] = \frac{\sigma}{\sqrt{\pi}}. \tag{17}$$

In this paper we make use of the following result giving the expected values of the pair of numbers (out of two pairs) that has the maximum sum. Suppose we have four normal random variables distributed as

$$\begin{aligned} A_1 &\sim \mathcal{N}(0,\sigma_A^2), \quad B_1 \sim \mathcal{N}(0,\sigma_B^2) \\ A_2 &\sim \mathcal{N}(0,\sigma_A^2), \quad B_2 \sim \mathcal{N}(0,\sigma_B^2). \end{aligned} \tag{18}$$

Let $\langle A_m, B_m\rangle$ be the $\langle A, B\rangle$ pair with the maximum sum. That is,

$$m = \underset{i\in\{1,2\}}{\operatorname{argmax}}\,(A_i + B_i). \tag{19}$$

The expected value of A_m is given by

$$E[A_m] = 2\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} a\phi_A(a)\phi_B(b)\Phi(a+b)\,\mathrm{d}a\,\mathrm{d}b, \quad (20)$$

$$\text{where } \Phi(a+b) = \int_{-\infty}^{a+b} \phi_{A+B}(c)\,\mathrm{d}c\,,$$

and $\phi_A(x)$ is the pdf of each of $A_{1,2}$, $\phi_B(x)$ is the pdf of each of $B_{1,2}$, and ϕ_{A+B} is the pdf of each of $A_1 + B_1$ and $A_2 + B_2$.

In words, we integrate over the possible values of A_1 and B_1, multiplying the joint probability of getting those values by the probability that the sum of the *other* $\langle A, B\rangle$ pair takes a value less than $A_1 + B_1$, and we multiply by the A_1 value to get its expected value. We then integrate over the possible values of A_2 and B_2 (for the case where the sum of the second pair is greater than the sum of the first), which gives the same integral again. Adding the two integrals together gives the expected value of A_m.

Evaluating the above integral gives the value

$$E[A_m] = \frac{\sigma_A^2}{\sqrt{\sigma_A^2 + \sigma_B^2}\sqrt{\pi}}, \quad (21)$$

and by symmetry the expected value of B_m is

$$E[B_m] = \frac{\sigma_B^2}{\sqrt{\sigma_A^2 + \sigma_B^2}\sqrt{\pi}}. \quad (22)$$

Acknowledgement

Thanks to Joe Mellor and Henry Reeve for their help in obtaining the result in the appendix. Thanks also to George Kourtis, Andrew Leeming, and Ubai Sandouk.

References

Alberch, P. (1991). From genes to phenotype: dynamical systems and evolvability. *Genetica*, 84(1):5–11.

Altenberg, L. (1994). The evolution of evolvability in genetic programming. In Kinnear, K. E., editor, *Advances in Genetic Programming*, Complex Adaptive Systems, pages 47–74, Cambridge. MIT Press.

Altenberg, L. (1995). Genome growth and the evolution of the genotype-phenotype map. In *Evolution and Biocomputation*, pages 205–259. Springer.

Baugh, D. and McMullin, B. (2013). Evolution of G-P mapping in a von Neumann self-reproducer within Tierra. In *Advances in Artificial Life, ECAL*, volume 12, pages 210–217.

Conrad, M. (1972). The importance of molecular hierarchy in information processing. *Towards a Theoretical Biology*, 4:222.

Dawkins, R. (2003). The evolution of evolvability. *On Growth, Form and Computers*, pages 239–255.

Ebner, M., Shackleton, M., and Shipman, R. (2002). How neutral networks influence evolvability. *Complexity*, 7(2):19–33.

Egri-Nagy, A. and Nehaniv, C. L. (2003). Evolvability of the genotype-phenotype relation in populations of self-replicating digital organisms in a Tierra-like system. In *Advances in Artificial Life*, pages 238–247. Springer.

Eiben, Á. E., Hinterding, R., and Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *Evolutionary Computation, IEEE Transactions on*, 3(2):124–141.

Gallagher, A. (2009). *Evolvability: a formal approach*. PhD thesis, University of Oxford.

Hansen, T. F. (2006). The evolution of genetic architecture. *Annual Review of Ecology, Evolution, and Systematics*, pages 123–157.

Hasegawa, T. and McMullin, B. (2013). Exploring the point-mutation space of a von Neumann self-reproducer within the Avida world. In *Advances in Artificial Life, ECAL*, volume 12, pages 316–323.

Kirschner, M. and Gerhart, J. (1998). Evolvability. *Proceedings of the National Academy of Sciences*, 95(15):8420–8427.

Maynard Smith, J. (1970). Natural selection and the concept of a protein space. *Nature*, 225:563–64.

McMullin, B. (2012). Architectures for self-reproduction: Abstractions, realisations and a research program. In *Artificial Life*, volume 13, pages 83–90.

Ofria, C. and Wilke, C. O. (2004). Avida: A software platform for research in computational evolutionary biology. *Artificial life*, 10(2):191–229.

Pelikan, M., Goldberg, D. E., and Lobo, F. G. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications*, 21(1):5–20.

Pigliucci, M. (2008). Is evolvability evolvable? *Nature Reviews Genetics*, 9(1):75–82.

Reisinger, J. and Miikkulainen, R. (2006). Selecting for evolvable representations. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pages 1297–1304. ACM.

Rudolph, G. (2001). Self-adaptive mutations may lead to premature convergence. *Evolutionary Computation, IEEE Transactions on*, 5(4):410–414.

Smith, T., Husbands, P., Layzell, P., and O'Shea, M. (2002). Fitness landscapes and evolvability. *Evolutionary Computation*, 10(1):1–34.

Stanley, K. O. and Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130.

Visser, J., Hermisson, J., Wagner, G. P., Meyers, L. A., Bagheri-Chaichian, H., Blanchard, J. L., Chao, L., Cheverud, J. M., Elena, S. F., Fontana, W., et al. (2003). Perspective: evolution and detection of genetic robustness. *Evolution*, 57(9):1959–1972.

Webb, A. M. and Knowles, J. (2014). Studying the evolvability of self-encoding genotype-phenotype maps. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 79–86. MIT Press.

Evolution of Cooperation in Evolutionary Robotics : the Tradeoff between Evolvability and Efficiency

Arthur Bernard[1,2], Jean-Baptiste André[3] and Nicolas Bredeche[1,2]

[1]Sorbonne Universités, UPMC Univ Paris 06, UMR 7222, ISIR, F-75005 Paris, France
[2]CNRS, UMR 7222, ISIR, F-75005, Paris, France
[3]Institut des Sciences de l'Evolution, Université de Montpellier, CNRS,
IRD, EPHE, CC065, Pl. E. Bataillon, 34095 Montpellier, France

Abstract

In this paper, we investigate the benefits and drawbacks of different approaches for solving a cooperative foraging task with two robots. We compare a classical clonal approach with an additional approach which favors the evolution of heterogeneous behaviors according to two defining criteria: the evolvability of the cooperative solution and the efficiency of the coordination behaviors evolved. Our results reveal a tradeoff between evolvability and efficiency: the clonal approach evolves cooperation with a higher probability than a non-clonal approach, but heterogeneous behaviors evolved with the non-clonal approach systematically show better fitness scores. We then propose to overcome this tradeoff and improve on both of these criteria for each approach. To this end, we investigate the use of incremental evolution to transfer coordination behaviors evolved in a simpler task. We show that this leads to a significant increase in evolvability for the non-clonal approach, while the clonal approach does not benefit from any gain in terms of efficiency.

Introduction

The evolution of cooperative actions in evolutionary robotics is as much a challenge as an interesting perspective for the design of complex collective systems (Doncieux et al., 2015). As such, it has been widely studied with very diverse approaches and objectives (Waibel et al., 2009; Hauert et al., 2014; Trianni et al., 2007; Lichocki et al., 2013). These works often use a clonal paradigm, where each robot has a copy of the same genome. This makes sense as this is the easiest way to ensure cooperation when individuals are expected to display similar behaviors. Moreover, using clones ensures maximal genetic relatedness between individuals, which is known to allow the evolution of altruism (Waibel et al., 2011; Montanier and Bredeche, 2011). As such, most research focus on increasing the probability for the cooperative solution to evolve.

In comparison, the nature of coordination behaviors and their influence on the quality of cooperation has yet to be thoroughly studied. In particular, interactions between clones in evolutionary robotics tend to produce homogeneous behaviors when most coordination tasks could benefit from heterogeneous behaviors. This could be solved by using a non-clonal approach where paired individuals do not use the same genome, and could possibly evolve different behaviors more easily. However, a non-clonal approach may face a *chicken-and-egg* dilemma: multiple individuals need to behave in a particular fashion for cooperation to be rewarding, but no benefit can be extracted from this behavior unless all individuals cooperate. Therefore, without cooperating partners, those behaviors cannot be selected by the evolution as they do not benefit the individual. This is particularly problematic when a moderately rewarding solitary strategy overshadows a more rewarding, but also more challenging to evolve, cooperative strategy (Skyrms, 2004).

In this paper, we are interested in the comparison between clonal and non-clonal approaches on two different criteria:

- *Evolvability* of cooperation, which is the number of successful runs where cooperation evolved.
- *Efficiency* of cooperation. This criteria is focused on the quality of the evolved behaviors and is determined by the performance (w.r.t. fitness score) of the coordination strategies.

To that end, we design a foraging task where both cooperative and solitary strategies are possible but where cooperation provides the largest reward. This task is favored by the evolution of efficient cooperative behaviors and we compare different approaches on both criteria. The first approach is a straightforward implementation of the literature where interacting individuals are clones. In comparison, the second approach is a rather extreme implementation of a non-clonal approach: we use coevolution, where individuals are from two different populations, and where fitness scores are computed independently for each individual. While this scheme is typical of competitive coevolution (Floreano and Nolfi, 1997; Floreano et al., 1998; Panait and Luke, 2005), the nature of the task considered here makes cooperation more interesting, as both individuals can selfishly benefit from being cooperative.

In the next section, we describe the methods and experimental setup used throughout our study. Then, we compare the results of the two approaches on the cooperative task. This first experiment reveals that both approaches face a tradeoff between evolvability and efficiency, where neither one dominates the other on both criteria. We investigate in a second experiment the possibility to overcome this tradeoff for both approaches. To this end, we use incremental evolution (Harvey et al., 1994; Urzelai et al., 1998) and evolve coordination in a simpler task in order to improve both the evolvability and efficiency on the target task for each approach. Finally, we discuss the implication of our findings in the last section, in particular with respect to maximizing evolvability and efficiency alike.

Methods

Two robotic agents are placed in a 800 by 800 units square arena with four solid walls and emptied from any obstacle apart from the targets in the foraging task. Each circular-shaped agent, with a diameter of 20 units, has a collection of sensors divided between a 90 degrees front camera and 12 uniformly distributed proximity sensors. The camera is composed of 12 rays with infinite range which indicate the type (coded on 3 bits) and proximity (one value in R^n) of the nearest object or agent in their direction. Proximity sensors have a range of twice the agent body's diameter and are used to get the distance to any obstacle nearby such as solid objects, the other agent or walls. The two agents always begin the simulation next to one another at one end of the arena, whereas all the objects' initial positions are randomized.

Agents can move freely in the environment and are controlled by a fully connected multi-layer perceptron with a single hidden layer, the topology of which does not change during the evolution. Inputs of this neural network are fed with all the data extracted from the sensors: 48 neurons for the camera (4 neurons for each of the 12 rays) and 12 neurons for the proximity sensors. A bias neuron, whose value is always 1, brings the total number of input neurons to 61. The hidden layer is comprised of 8 neurons and the output layer of 2 neurons giving the speed of each of the agent's wheels. The activation function used is a sigmoid.

In each experiment, individuals evolved during a fixed amount of evaluations thanks to an evolutionary algorithm. Their genome consists of a collection of the 506 connection weights (as real-values) of the neural network and is initially randomized for each individual in the population. Three evaluation setups are used to compare the different approaches of our experiment:

- In the *control* setup, each individual is evaluated against 5 other randomly chosen individuals in the population except itself. Therefore we ensure that there is no genetic relatedness between individuals in each pair. However, it is not clear how the evolutionary algorithm itself may impact the population's diversity, especially because elitism is used;

- In the *clonal* setup, each individual is evaluated once against a clone of itself. This setup is used to study the results of the classical clonal approach (Waibel et al., 2009; Hauert et al., 2014; Trianni et al., 2007; Lichocki et al., 2013). While previous works have shown on multiple occasions that cooperation can evolve, it is not clear if individuals can take different roles during a cooperative interaction;

- In the *coevolution* setup, each of the two individuals comes from two different coevolved populations. In this setup, each individual from one population encounters 5 random individuals from the other population. As pairing considers individuals from two seperate populations, the evolution of heterogeneous behaviors is theoretically easier. As a matter of fact, such a relation where two very different individuals find a selfish interest in mutual cooperation is actually quite common in nature (Connor, 1995).

A pair of individuals then interact in the arena described before for a fixed number of simulation steps called a trial. Each trial is conducted 5 times to account for the random initial positions of the objects and decrease the influence of the initial conditions on the individuals' performance.

The selection method used in the evolutionary algorithm is an elitist (10+10)-ES where the 10 best individuals in the population are used to generate 10 offsprings for the next generation. We use no recombination and therefore each offspring is a mutated copy of its parent. Mutations were sampled according to a gaussian operator with a standard deviation of 1.10^{-2} and a gene's mutation rate of 5.10^{-3}. Finally, population size was kept constant through the evolution with a number of 20 individuals. All experiments were done using the framework for evolutionary computation SFERESv2 (Mouret and Doncieux, 2010), which includes a fast 2D robotic simulator. The source code for reproducing the experiments is available for download at http://pages.isir.upmc.fr/˜bredeche/Experiments/ECAL2015-coop.tgz.

Cooperative Foraging Task

In this first experiment, we investigate the evolution of cooperation in a foraging task. The environment is filled with 18 solid targets that the agents can collect. To collect a target, an agent has to stay close to this object for a fixed amount of simulation steps (800). After this duration, the target disappears and any agent close to it is rewarded with its value. Targets are of two types. Green targets always reward 50 when collected whereas purple ones reward 250 only when the agents collect it together (Table 1). If a solitary agent collects a purple target, it disappears and rewards nothing. Consequently, there is both an incentive and a risk to cooperate as cooperation is dependent on successful coordination. This setup is a robotic implementation of a well-known problem in game theory for studying the evolution of mutualistic cooperation: the *Stag Hunt* (Skyrms, 2004).

The fitness score (F) of an individual is the average reward per trial:

$$F = \frac{1}{N * M} \sum_{i=1}^{N} \sum_{j=1}^{M} f_{ij}$$

Where N is the number of individuals encountered (5 in the control and coevolution setups, 1 in the clonal setup), M the number of trials (5) and f_{ij} the rewards obtained at trial j with individual i.

When a target is collected, another target of the same type is then placed at a random position in the arena to keep a constant ratio between green and purple targets. Each evaluation lasted 20000 simulation steps and 60 independent runs were conducted for each experimental setup, each one lasting 40000 evaluations.

Target		Reward
Green		
	alone	50
	coop	50
Purple		
	alone	0
	coop	250

Table 1: Rewards for the foraging of the different targets, depending on whether they were collected alone or cooperatively.

Results

Setting	# Coop.	# Solitary	Total
Control	10	50	**60**
Clonal	28	32	**60**
Coevolution	14	46	**60**

Table 2: Number of simulations where the best individual evolved a cooperative strategy (collecting purple targets) or a solitary strategy (collecting green targets) for each setup in the foraging task.

We are interested in the number of simulations where cooperation evolved (i.e. the *evolvability* of each approach), which means simulations where the best individual in the population evolved the cooperative foraging of the purple targets (i.e. more than 50% of the collected targets are purple). Results for the three setups are displayed in Table 2. As could be expected from the literature, the clonal setup displays a greater evolvability w.r.t. evolving cooperation (28/60), whereas coevolution (14/60) is on par with the control setup (10/60). It is also apparent that cooperation is still difficult to evolve as in the best case (clonal), no more than half the simulations display the evolution of cooperative behaviors.

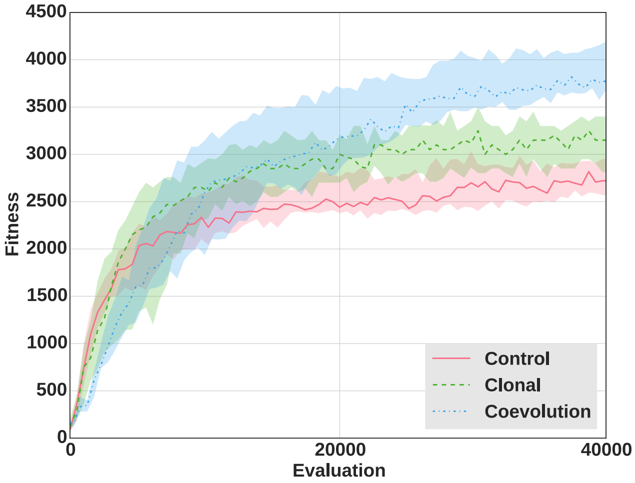


Figure 1: Median fitness score of the best individuals in each of the runs where cooperation evolved for each setup over time. The fitness score of an individual is computed as the average reward the individual earned per trial by foraging targets. The colored areas around the medians represent the first and third quartiles.

However, cooperative individuals do not perform with the same *efficiency* from one setup to another. We show in Figure 1 the median fitness score of the best individuals in each independent run where cooperation evolved over time and for each setup. Fitness scores are significantly different in each setup with the best score obtained in the coevolution setup and the worst in the control setup (Mann-Whitney U-test on the fitness score of the best individuals at the last evaluation, p-value < 0.001).

These differences in efficiency can be explained by looking at the nature of the cooperative behaviors evolved, which reveals two types of behaviors: *turning* and *leader/follower*.

Individuals adopting the turning strategy turn around one another so that they always see the other individual as well as stay close to it (Figure 2(a)). This allows the two individuals to approach simultaneously a same target and therefore forage it in a cooperative fashion. In this strategy, both individuals have a similar behavior and no role division is necessary for their successful cooperation.

In comparison, individuals which evolve a leader/follower strategy adopt a differentiation between two roles: *leader* and *follower* (Figure 2(b)). The individual we call leader always goes first on a target whereas the follower always arrives second on the same target. We observe that the follower's behavior consists in staying close to the leader and always keeping it in front of itself. In comparison the leader shows a lesser interest in the presence of its follower and rarely checks on its position.

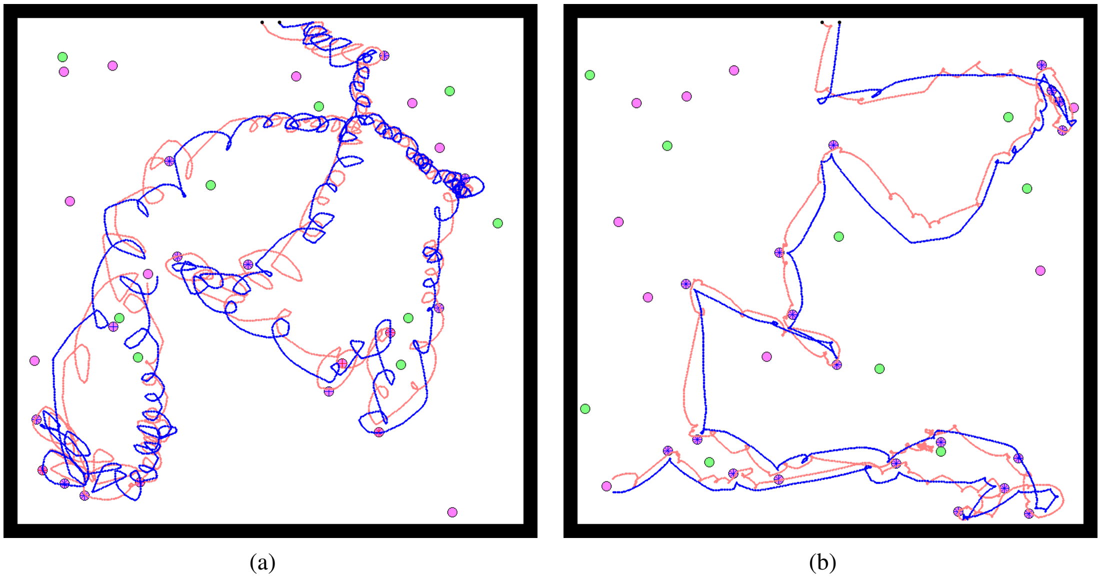


Figure 2: Snapshots of the simulation after an entire trial in the foraging task. The path of each robotic agent from their initial positions (black dots) is represented in red and blue. The green and purple discs represent the 18 targets in the environment. When a target is foraged by the two agents, a red cross (resp. blue) is drawn on the target if the red agent (resp. blue) arrived on it first. Each snapshot corresponds to a trial where agents adopted a different behavior: *(a)* turning or *(b)* leader/follower.

Table 3 shows the distribution of cooperative strategies for all three setups. Whereas the control and clonal setups always resulted in turning strategies (resp. 10/10 and 28/28), the coevolution setup always displayed the evolution of a leader/follower strategy (14). We observe that this latter strategy leads to more efficient cooperation. Indeed, individuals adopting the turning strategy are forced to check constantly on the other individual's position. Consequently, they cannot be as fast as individuals with a leader/follower strategy where they move to the target in a straight line under the leader's direction. Moreover, due to the random proximity of the targets, the turning strategy is prone to errors. Namely, they often get to another target than that of their partner whenever two targets are too close to each other.

A possible explanation as to why no leader/follower strategy could evolve in the control and clonal setups may be because of the need to differentiate between the two roles. Indeed, there needs to be the existence of an asymmetry between the two individuals for this phenomenon to appear. With coevolved populations, this asymmetry is deliberately created by the separation between the two populations. Indeed, we observe that one population exclusively contains leaders while the other exclusively contains followers.

The two other setups fail to evolve heterogeneous behaviors. In the control setup, this may be due to the evolutionary algorithm used, especially with elitism enforcing the homogenization of the population throughout the course of evolution (as hinted in the Methods Section). Then, the clonal setup introduces yet another challenge as switching to a particular role can only be done during evaluation as both individuals are by definition genetically similar.

Setting	**# Leader/Follower**	**# Turning**	**Total**
	Strategy	**Strategy**	
Control	0	10	**10**
Clonal	0	28	**28**
Coevolution	14	0	**14**

Table 3: Repartition of the different strategies evolved in each of the runs where cooperation evolved for each setup in the foraging task. We indicate in each cell the number of simulations where a particular strategy evolved.

Going Beyond the Evolvability vs. Efficiency Tradeoff using Incremental Evolution

The previous section revealed a tradeoff between evolvability and efficiency. In the clonal setup, cooperation evolves more often than with other setups. However, the coevolution setup yields cooperative behaviors which are more efficient, with paired individuals displaying asymmetrical behaviors.

In this section, we address the following question: is it possible to benefit from both evolvability *and* efficiency with the clonal and/or the coevolution setups? In other words, we explore (1) whether the clonal setup can be used to evolve pairs with heterogeneous behaviors, and (2) whether the coevolution setup can be improved in terms of number of runs where cooperation evolved.

In order to address this question, we use incremental evolution, a rather common method in evolutionary robotics for solving challenging problems (Dorigo and Colombetti, 1994; Saksida et al., 1997; Bongard, 2008; Doncieux, 2013). The main principle is to ease the learning of a complex task by splitting it into simpler sub-tasks (Perkins and Hayes, 1996).

In the following, we introduce an additional task, the *waypoints crossing* task, which requires the evolution of coordination behaviors, and is simpler to address than the previous task. Individuals evolved in this first task are then used as starting point for the original task described earlier, hoping that cooperative behavior will be *recycled* from the first task to the second task.

Waypoints Crossing Task

We consider a task where robotic agents have to cross randomly positioned waypoints. As such, these round waypoints do not act as obstacles and have a diameter of 30 units. As soon as an agent goes through a waypoint, it can not be seen by this agent anymore. All 18 waypoints have the same color and can be crossed in any order. The fitness score (F) of each individual is defined as the average longest sequence of waypoints shared by both agents per trial:

$$F = \frac{1}{N * M} \sum_{i=1}^{N} \sum_{j=1}^{M} l_{max_{ij}}$$

Where N is the number of individuals encountered (5 in the control and coevolution setups, 1 in the clonal setup), M the number of trials (5) and $l_{max_{ij}}$ the longest sequence of waypoints shared by both individuals at trial j with individual i.

This implies that the two individuals are rewarded when crossing waypoints in the same order as well as maximizing the number of waypoints crossed. Each evaluation lasted 10000 simulation steps and 60 independent runs were conducted for each experimental setup, each one lasting 40000 evaluations.

All simulations showed an increase in fitness score for each of the three setups (cf. Figure 3). This was expected as this task does not represent a particular challenge for the individuals: it simply needs the evolution of a successful coordination strategy. However, whereas the coevolution and clonal setups performed equally, they both surpassed the performance of individuals from the control setup (Mann-Whitney, p-value < 0.001).

As with the previous foraging task, we can hypothesize that these differences in fitness scores are due to differences in the behaviors evolved. Table 4 gives a classification of the cooperative behaviors for each setup. They are similar to those in the previous task with the addition of a third rare strategy: the *wall-following* strategy (which is regrouped in "Other"). Wall-followers simply follow the walls around the arena and cross any waypoints close to the wall they are adjacent to. As such, this is a far less efficient strategy than the two others.

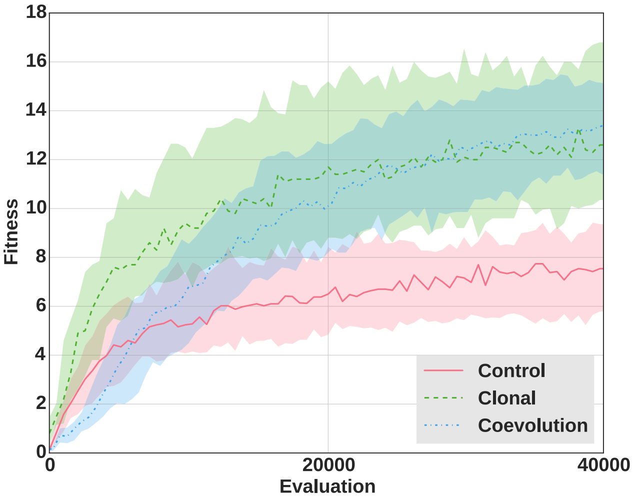
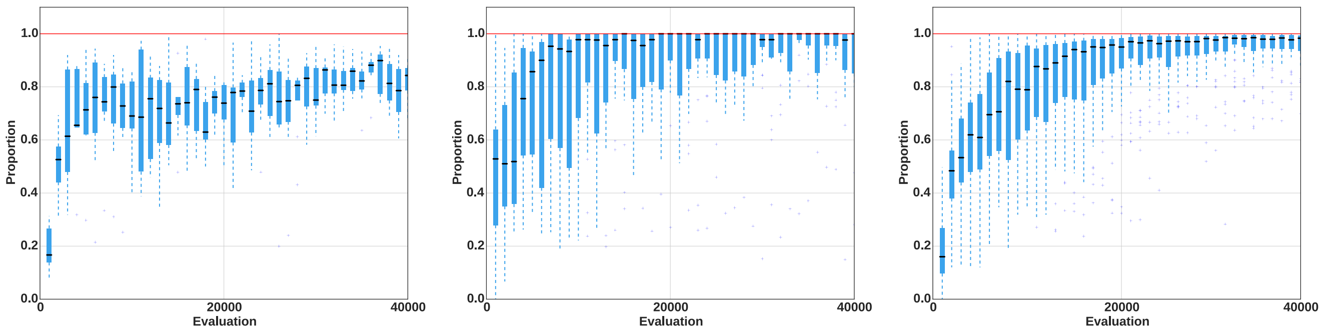


Figure 3: Median fitness score of the best individuals in each of the 60 independent runs and for each setup over time. Fitness score is computed as the average longest sequence of waypoints shared by both agents per trial. The colored areas around the medians represent the first and third quartiles.

Setting	# Lead.	# Turn.	# Other	Total
Control	19	37	4	**60**
Clonal	23	31	6	**60**
Coevolution	59	1	0	**60**

Table 4: Repartition of the different strategies evolved in each of the 60 independent runs for each setup in the waypoints task. We indicate in each cell the number of simulations where a particular strategy evolved: *Leader/follower* (Lead.), *Turning* (Turn.) or *Other*. "Other" regroups wall-following strategies or simulations where no recognizable strategy evolved.

In the coevolution setup, nearly all runs (59/60) evolved a leader/follower strategy. Interestingly, although fitness scores in the clonal and control setups are significantly different, this behavior evolved in roughly one third of the runs for both setups. To explain the difference in fitness scores, we must take into account the quality of the leader/follower strategy in each setup. We measure the proportion of leadership in each interaction, which is computed as the proportion of waypoints crossed by both individuals for which the leader arrived first. Figure 4 displays the boxplots of the proportion of leadership for the best individuals in each setup and only for the simulations where a successful leader/follower strategy evolved (a minimal threshold of 0.75 is chosen to consider only the best performing runs).



(a) (b) (c)

Figure 4: Boxplots of the proportion of leadership over time for the best individuals in each runs where the proportion at the last evaluation was greater than 0.75 in the *(a)* control, *(b)* clonal or *(c)* coevolution setup. This value represents the proportion of waypoints crossed by both individuals for which the leader arrived first.

We show that the proportion of leadership is greater in the clonal and coevolution setups than in the control one (Mann-Whitney, p-value < 0.005). These differences means that the individuals are more efficient in their leader/follower strategy in the clonal and coevolution settings than in the control one. This explains the differences in fitness scores observed in Figure 3.

Interestingly, whereas in the foraging task no leader/follower strategy could evolve in the control and clonal setups, this strategy did evolve in one third of the simulations for this task. This could mean that these individuals use information in the environment to adopt one role or the other. Indeed, we observe that this is achieved by exploiting the differences in the initial starting positions, with one individual on the left and the other on the right. They both turn to the same direction (left or right, depending on the runs) at the beginning of the simulation which results in one individual (the leader) turning its back to the other, while the second individual (the follower) looking at its partner.

Recycling Cooperative Behaviors in the Foraging Task

Coming back to the initial foraging task, we perform the exact same experiment described at the beginning of this paper, with one notable exception: the initial population is initialized with genomes evolved for solving the waypoint task. This implies that coordination is possible starting from the very first generation of each setup. Given that we have already shown that such coordination is a desirable feature, the question is: will it be possible to retain cooperative behaviors in order to solve the foraging task?

Table 5 gives the results in terms of evolved behaviors from the 60 independent runs for each setup. The coevolution setup evolves cooperation slightly more often (28/60) than both the control (20/60) and the clonal (24/60) setups. A first remark is that the number of occurences of cooperation for the coevolution and control setups have actually doubled compared to previous results without incremental evolution (see Table 2). This is not the case for the clonal setup, which does not appear to benefit from incremental evolution.

Setting	**# Coop.**		**# Solitary**	**Total**
	# Lead.	**# Turn.**		
Control	0	20	40	**60**
Clonal	0	24	36	**60**
Coevolution	28	0	32	**60**

Table 5: Proportion of the 60 independent simulations where the best individual evolved a cooperative strategy (collecting purple targets) or a solitary strategy (collecting green targets) for each setup in the foraging task when individuals are previously evolved in the waypoints task. In addition, the repartition of the different strategies is indicated when cooperation evolved: *Leader/Follower* (Lead.) or *Turning* (Turn.).

A second remark is that cooperation in the coevolution setup systematically corresponds to a leader/follower strategy, which is *never* the case with the two other setups. This has a significant, though expected, impact on fitness scores, as shown in Figure 5. Cooperation evolved with the coevolution setup leads to significantly greater fitness scores (Mann-Whitney, p-value < 0.001).

Results from this experiment make it possible to revise our initial statement. Using pre-trained individuals strongly benefits the coevolution setup in terms of evolvability. But this is not the case with the clonal setup, for which using pre-trained individuals improves neither evolvability nor efficiency. Therefore, we may face a tradeoff which does not concern evolvability and efficiency, but one that implies computational cost: the coevolution setup outperforms the clonal setup on both evolvability and efficiency *at the cost* *of additional computational effort*.

The control and clonal setups completely failed to maintain a leader/follower strategy, even though such strategy originally evolved. An explanation is provided by considering the difference between the waypoints task, where leader/follower evolved, and the current foraging task. In the waypoints task, symmetry breaking could be achieved at the beginning of the evaluation (as explained earlier), and could be retained afterwards as the follower was always behind the leader. However, the current foraging setup requires that the two robots display the same behavior to cooperatively collect a target (ie. both robots have to touch the target), which implies that leader/follower roles are lost, as they depend on the relative position of robots with one another.

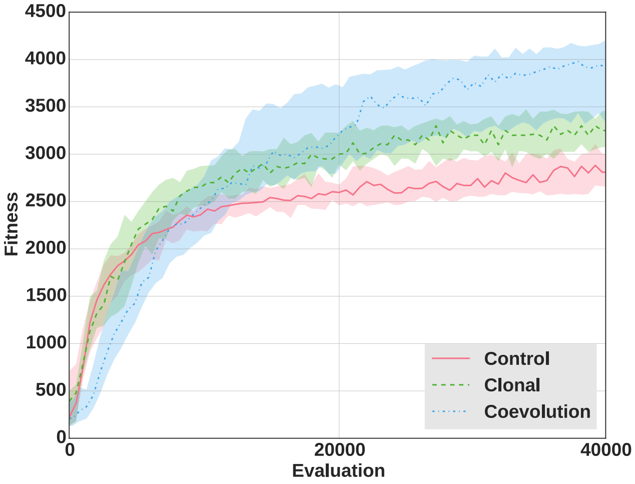


Figure 5: Median fitness score of the best individuals in each of the runs where cooperation evolved for each setup over time. The fitness score of an individual is computed as the average reward the individual earned per trial by foraging targets. The colored areas around the medians represent the first and third quartiles.

Discussion and Conclusion

In this paper, we considered several approaches for the evolution of cooperation in evolutionary robotics: a clonal approach, where all individuals in a group share the same genome, and a non-clonal approach, where individuals are independent from one another, but may share a common interest in cooperating.

We first showed that there exists a tradeoff between evolvability and efficiency. On the one hand, the clonal approach evolves cooperative behaviors on a more frequent basis than with the other approach. On the other hand, the non-clonal approach, which is implemented using a coevolution setup, results in more efficient behaviors in terms of pure performance whenever cooperation evolved. The non-clonal approach actually enables the evolution of asymmetric behaviors, such as a leader/follower strategy.

We then used incremental evolution to evolve coordination behaviors using a simpler task in order to overcome this tradeoff and improve both evolvability and efficiency in each setup. We showed that while no improvement was observed in the clonal setup on either criteria, the outcome is very different for the coevolution setup: the probability of evolving cooperation actually increases, and the evolved cooperative solutions remain the most efficient.

This work raises several questions. Firstly, heterogeneous behaviors were obtained with coevolution, a rather radical way to enable asymmetrical behaviors during cooperation. However, the waypoints task revealed that breaking symmetry can also be done with identical individuals using environmental feedback, even though such cooperation is difficult to obtain. As a consequence, we intend to investigate the evolution of cooperation with heterogeneous behavior without resorting to coevolution. In particular, we will study how more elaborated neural architectures (e.g. using plasticity) can switch to a particular persistant regime depending on environmental cues available at the beginning of the evaluation.

Secondly, incremental evolution requires an added computational cost in order to increase evolvability in the non-clonal approach. However, it may be possible to avoid this extra cost by considering other evolutionary methods. In particular, we intend to explore how a multiobjective approach which considers both performance and *diversity* could improve the optimization process (Lehman and Stanley, 2008; Doncieux and Mouret, 2014). Though this approach looks promising, it is not clear yet how diversity should be implemented in the context of cooperative problem solving.

Acknowledgements

This work is supported by the ANR CreAdapt project (ANR-12-JS03-0009) and the European Union's Horizon 2020 research and innovation programme under grand agreement No 640891. Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

Bongard, J. (2008). Behavior Chaining : Incremental Behavior Integration for Evolutionary Robotics. In *Artificial Life XI: Proceedings of the 11th International Conference on the Simulation and Synthesis of Living Systems*, pages 64–71. MIT Press.

Connor, R. C. (1995). The Benefits of Mutualism: A Conceptual Framework. *Biological Reviews*, 70(3):427–457.

Doncieux, S. (2013). Transfer Learning for Direct Policy Search: A Reward Shaping Approach. In *Proceedings of the International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pages 1–6. IEEE.

Doncieux, S., Bredeche, N., Mouret, J.-B., and Eiben, A. E. (2015). Evolutionary Robotics: What, Why, and Where to. *Frontiers in Robotics and AI*, 2(4).

Doncieux, S. and Mouret, J.-B. (2014). Beyond black-box optimization: a review of selective pressures for evolutionary robotics. *Evolutionary Intelligence*, pages 1–18.

Dorigo, M. and Colombetti, M. (1994). Robot Shaping : Developing Autonomous Agents through Learning. *Artificial Intelligence*, 70(2):321–370.

Floreano, D. and Nolfi, S. (1997). God Save the Red Queen! Competition in Co-Evolutionary Robotics. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 398–406.

Floreano, D., Nolfi, S., and Mondada, F. (1998). Competitive Co-Evolutionary Robotics: From Theory to Practice. In *From Animals to Animats 5: Proceedings of the 5th International Conference on Simulation of Adaptive Behavior*, pages 515–524.

Harvey, I., Husbands, P., and Cliff, D. (1994). Seeing The Light : Articial Evolution, Real Vision. In *From Animals to Animats 3: Proceedings of the 3rd International Conferance on Simulation of Adaptive Behavior*, volume 1994, pages 392–401.

Hauert, S., Mitri, S., Keller, L., and Floreano, D. (2014). Evolving Cooperation : From Biology to Engineering. In *The Horizons of Evolutionary Robotics*, pages 203–217. MIT Press.

Lehman, J. and Stanley, K. O. (2008). Exploiting Open-Endedness to Solve Problems Through the Search for Novelty. *Artificial Life XI*, pages 329–336.

Lichocki, P., Wischmann, S., Keller, L., and Floreano, D. (2013). Evolving team compositions by agent swapping. *IEEE Transactions on Evolutionary Computation*, 17(2):282–298.

Montanier, J.-M. and Bredeche, N. (2011). Surviving the Tragedy of Commons : Emergence of Altruism in a Population of Evolving Autonomous Agents. In *Proceedings of the 11th Internation Conference on Artificial Life (ECAL'11)*, pages 550–557.

Mouret, J.-B. and Doncieux, S. (2010). SFERESv2: Evolvin' in the Multi-Core World. In *Proceedings of Congress on Evolutionary Computation (CEC)*, pages 4079–4086. IEEE.

Panait, L. and Luke, S. (2005). Cooperative Multi-Agent Learning: The State of the Art. *Autonomous Agents and Multi-Agent Systems*, 11(3):387–434.

Perkins, S. and Hayes, G. (1996). Robot Shaping - Principles, Methods and Architectures. Technical report, University of Edinburgh, Edinburgh, GB.

Saksida, L. M., Raymond, S. M., and Touretzky, D. S. (1997). Shaping Robot Behavior using Principles from Instrumental Conditioning. *Robotics and Autonomous Systems*, 22:231–249.

Skyrms, B. (2004). *The Stag Hunt and the Evolution of Social Structure*. Cambridge University Press.

Trianni, V., Ampatzis, C., Christensen, A., Tuci, E., Dorigo, M., and Nolfi, S. (2007). From Solitary to Collective Behaviours: Decision Making and Cooperation. In Almeida e Costa, F., Rocha, L., Costa, E., Harvey, I., and Coutinho, A., editors, *Advances in Artificial Life*, volume 4648 of *Lecture Notes in Computer Science*, pages 575–584. Springer Berlin Heidelberg.

Urzelai, J., Floreano, D., Dorigo, M., and Colombetti, M. (1998). Incremental Robot Shaping. *Connection Science*, 10(3-4):341–360.

Waibel, M., Floreano, D., and Keller, L. (2011). A Quantitative Test of Hamilton's Rule for the Evolution of Altruism. *PLoS biology*, 9(5):e1000615.

Waibel, M., Keller, L., Floreano, D., and Member, S. (2009). Genetic Team Composition and Level of Selection in the Evolution of Cooperation. *IEEE Transactions on Evolutionary Computation*, 13(3):648–660.

An Investigation of Square Waves for Evolution in Carbon Nanotubes Material

Odd Rune Lykkebø[1], Stefano Nichele[1] and Gunnar Tufte[1]
[1]Norwegian University of Science and Technology, Trondheim, Norway
{lykkebo, nichele, gunnart}@idi.ntnu.no

Abstract

Materials suitable to perform computation make use of evolved configuration signals which specify how the material samples are to operate. The choice of which input and configuration parameters to manipulate obviously impacts the potential of the computational device that emerges. As such, a key challenge is to understand which parameters are better suited to exploit the underlying physical properties of the chosen material. In this paper we focus on the usage of square voltage waves as such manipulation parameters for carbon nanotubes/polymer nanocomposites. The choice of input parameters influence the reachable search space, which may be critical for any kind of evolved computational task. We provide common measurements such as power spectrum and phase plots, taken with the the Mecobo platform, a custom-built board for evolution-in-materio. In addition, an initial investigation is carried out, which links the frequency of square waves to comparability of the output from the material, while also showing differences in the material's physical parameters. Observing the behaviour of materials under varying inputs allows macroscopic modelling of pin-to-pin characteristics with simple RC circuits. Finally, SPICE is used to provide a rudamentary simulation of the observed properties of the material. This simulation models the per-pin behaviours, and also shows that an instance of the traveling-salesman-problem can be solved with a simple randomly generated cloud of resistors.

Introduction and Background

Evolution-in-Materio (EIM) (Miller et al., 2014), (Miller and Downing, 2002) is a bottom-up approach where the intrinsic underlying physics of materials is exploited as computational medium. In contrast to a traditional design process where a computational substrate, e.g. silicon, is precisely engineered, EIM uses a bottom-up approach to manipulate materials with the aim of producing computation. Such manipulation is done with computer controlled evolution (CCE) (Harding and Miller, 2007), (Harding et al., 2008). CCE may program the materials with different kinds of stimuli, e.g. voltages and currents, temperature, and magnetic fields.

In the NASCENCE project (Broersma et al., 2012), novel nano-scale materials are being used as a substrate in which computation is attempted. In particular carbon nanotubes (CNTs) / polymer have shown promising for the solution of Travelling Salesman (Clegg et al., 2014), logic gates (Kotsialos et al., 2014), and function optimization problems (Mohid et al., 2014). To solve problems, the material is required to hold physical richness (Miller et al., 2014) under a certain manipulation scheme. The method used to manipulate the material into doing computation has until now largely consisted of setting up input signals of different kinds, e.g. static voltages (Clegg et al., 2014), square wave voltages (Lykkebø et al., 2014), a mix of both (Lykkebø and Tufte, 2014), across gold plated connectors that are exposed to the material via a glass plate. Square waves of different frequencies demonstrated potential to achieve a computationally rich behaviour (Nichele et al., 2015). As such, they are the main subject of investigation in this paper.

The hypothesis is that this electrical current exploits properties in the material that gives rise to a potentially useful output, i.e. there is an emergent behaviour emanating from interactions between current travelling through the provably non-linear material (in terms of the current/voltage relation (Massey et al., 2011) and jumps in conductivity under certain temperatures and geometric variation (Ebbesen et al., 1996)) and that measurements of this behaviour can be used for computation.

Is the material performing this computation? It can be very hard to pinpoint what computation actually is and where it happens. One key requirement is Ashby's requisite variety (Ashby, 1956), in which the importance of how many states the system can be in is underlined. In the context of computation in materials, the number of possible input states needs at least to have correspondence in physical states the material can be in. It is worth noting the difference between observable states and unobservable states. The number of observable states can be much lower than the number of states that the material can be in. When manipulating the material with an input, e.g. square waves, the material can iterate through a number of such internal states before eventually settling on the final emergent observable state, which can be in many forms, such as varying voltage peaks or phase offsets. Intuitively, these properties lead to an output of varying complexity, and an intuitively attractive idea is that the measured complexity of the output gives an indication about the internal states of the material as well; if the measured output is far more complex than the inputs, there must necessarily be a mechanism (computation) that gives rise to this complexity.

What is the best way of exploiting such computational properties? Our hypothesis is that materials with high CNT densities may act more as a conductive layer. With lower CNT densities, conductive paths may be closer to the electric percolation threshold of the CNT network (around 1% nanotubes). Another aspect that impact on computational properties is the selected range of frequencies for input signals. Specific frequencies may better penetrate the material as a result of the exploited CNT paths and signal feedbacks. Our aim is to gather basic knowledge of different material's computational properties as to be able to create a simple material model based on RC electrical circuits.

In the experiments herein, material samples with different concentrations of SWCNTs are investigated using square waves at different frequencies. Signal outputs are measured in terms of power spectrum and phase plots. In addition, compressibility of the output signals is investigated for different input frequencies and number of input pins, i.e. number of input frequencies. Such compressibility measure is close to Kolmogorov approximations (Kolmogorov, 1965), (Nichele and Tufte, 2013) of signal complexity.

Observing the response of materials with different concentrations of SWCNTs to given inputs indicate that a simple model of materials can be based on RC circuits. Such a model is developed using SPICE. The presented results provide initial thoughts regarding computational properties of used materials, together with aspects that need to be taken into account when tackling computational problems using evolution-in-materio, i.e. stability, repeatability, and noise.

The article is laid out as follows: Section II describes setup and experimental method. Section III presents the experimental results. In Section IV the modelling using SPICE is detailed and Section V provides a discussion, conclusion and ideas for further work.

Setup and Methodology

The material used for this work, i.e. carbon nanotubes/polymer nanocomposite, is placed on a micro-electrode array on a glass slide which is inserted into the Mecobo board. Each sample has 16 electrodes which are physically connected to the board and can be stimulated by electrical signals, e.g. static voltages, square waves. A sketch of the experimental setup is shown in Figure 1. See (Lykkebø et al., 2014) for more details. To investigate parts of the characteristics of the material in our system under the influence of various frequencies, we run a simple frequency analysis of each material, doing a Fourier transform and find the power spectrum and phase spectrums for 3 different material samples, and at 4 different frequencies: 1KHz, 10KHz, 50KHz, 1MHz (for space reasons and to try a broad range, relative to what the board can output). The tested materials are shown in Table 1.

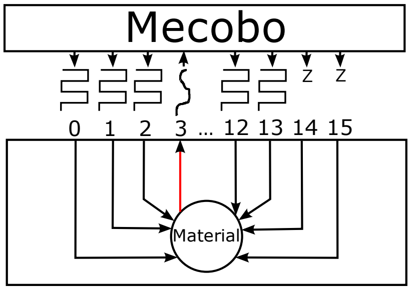


Figure 1: Experimental setup

Material	SWCNT Concentration	DC resistance
B15S01	0.75%	20k Ω
B15S02	1.00%	5k Ω
B15S05	1.75%	1k Ω

Table 1: Parameters of tested materials

All measurements were taken with the Mecobo board as described in (Lykkebø et al., 2014). Pin 3 is used as sampling pin on all measurements, and the samples were taken at 500KHz for 50 ms, thus collecting 20,000 samples.

For the compression tests (see next chapter for details), pin 3 was used as a sampling pin and input pins *to* the material were added incrementally starting at pin 0 and increasing to pin 15, skipping pin 3. We also increased the number of tested frequencies since these plots were easier to visually compress and present in this paper. The number of samples collected is deterministic. After collecting the samples, they were converted to a binary string and appended to a growing string of bits of total length 800,000 (32 bits per float), which was then passed to the built-in Python 2.7 compress()-function. The length of the compressed buffer was then measured and plotted.

Results and discussion

This Section outlines two sets of experiments. The first part deals with investigating how materials with different concentrations of SWCNTs behave when exposed to square waves at different frequencies. Output voltages, Fourier transform and phase plots are compared. In the second part, the material samples with different nanotube concentrations are analyzed again in terms of compressibility of output. As such, we may be able to relate input frequencies and output complexity in terms of compressibility for different materials.

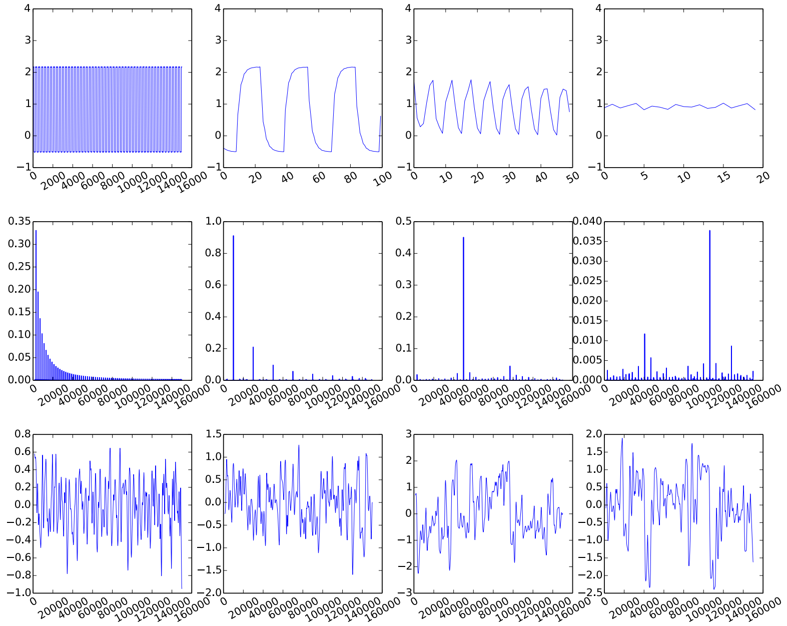


Figure 2: Material B15S01(0.75%) - Each column is a different frequency (1KHz, 10KHz, 50KHz, 1MHz). Row 1; raw voltage/time, row 2; power spectrum, row 3; phase spectrum.

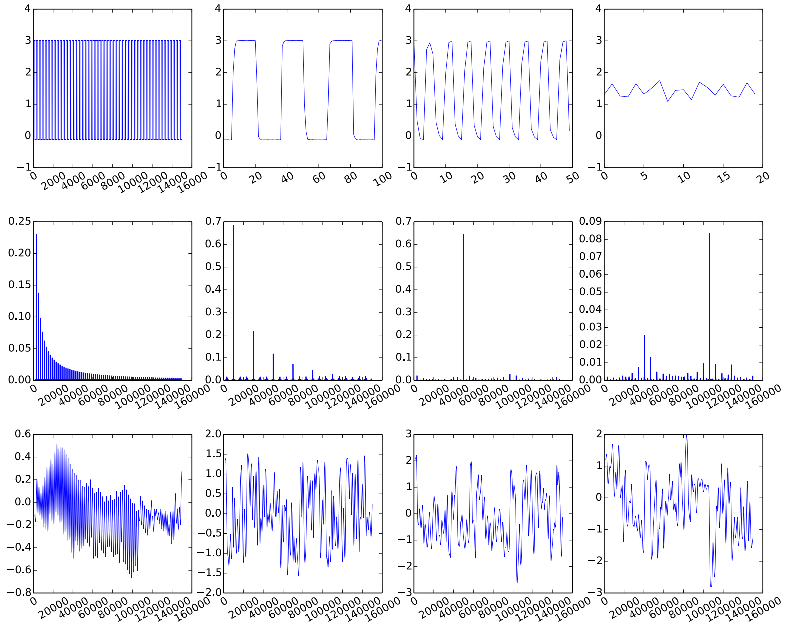


Figure 3: Material B15S02(1.00%) - Each column is a different frequency (1KHz, 10KHz, 50KHz, 1MHz). Row 1; raw voltage/time, row 2; power spectrum, row 3; phase spectrum.

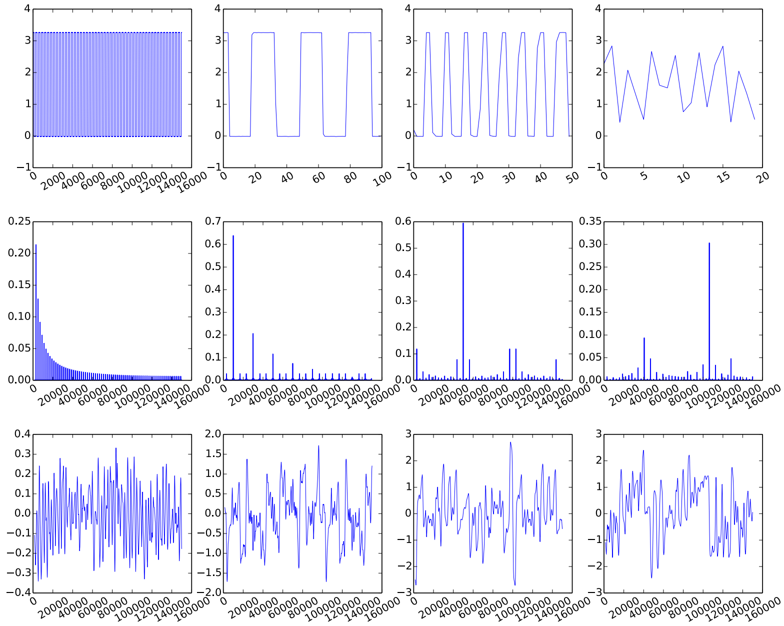


Figure 4: Material B15S05(1.75%) - Each column is a different frequency (1KHz, 10KHz, 50KHz, 1MHz). Row 1; raw voltage/time, row 2; power spectrum, row 3; phase spectrum.

Basic frequency measurements

We will first discuss Figures 2, 3 and 4. Each column corresponds to a different frequency. As can be seen from the first row in the figures, the peak-to-peak amplitude goes down as we apply higher frequencies. We believe this is the effect of capacitance in the material.

When sweeping square wave frequencies from 1Hz to 1MHz, the material initially passes the signal through (low pass), and eventually starts to let the high frequency through as well (high pass). Applying Occam's razor, the easiest way to explain this behaviour is to assume that the material has a certain amount of capacitance in addition to the easy-to-measure DC resistance. This capacitance makes the material exhibit a charge/discharge cycle when exposed to a time-varying signal such as a square wave. Another possibility is that the inductance of the material is responsible for the filtering effects. Inductance, however, is due to winding of a conductor and the distribution of the CNTs in the PMMA does not immediately appear to be laid out in a such a way as to give rise to this phenomena. Since inductive impedance increases with frequency by $j\omega L$, where L is inductance, a relatively high inductance value is required to follow the curve of the signal, as seen in Figure 12. This is also the reason why the model described below does not include any inductors; they do not seem to contribute any major factor at this observation level and at the chosen range of frequencies to investigate.

Moving on to row 2 in Figures 2, 3, 4, which shows the power spectra for each material reveals a couple of things. First, it should be noted that we have removed the DC Fourier frequency bin from the plots. Secondly, the plots show that the platform is relatively free of obvious noise. The biggest frequency responses come from the applied signals, as expected. Notice by studying the y-axis of the plots that the power goes down significantly as we increase the frequency, as is expected since the peak-to-peak voltage is also rapidly decreasing. The high-frequency plots do seem to have several more frequencies present, but this is simply an artifact of the axis scaling.

The last row in Figures 2, 3 and 4, shows the phase information from the Fourier transform. Although harder to interpret intuitively, the relatively noisy response shows that there are no obvious trends in the phase of the signals across the frequency range, which means that it is little use in trying to infer any computational properties that make use of the phase information.

Compression tests

The results from the compression test can be seen in Figures 5, 6 and 7. Note that the standard error is quite low. The main observation from these graphs is that as we add more pins that output current *into* the material, the length of the

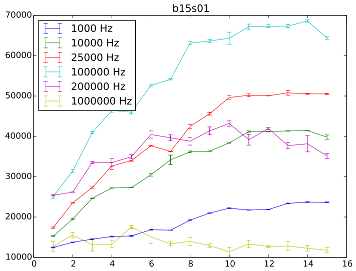


Figure 5: Material B15S01, length of compressed buffer vs. increasing number of input pins enabled.

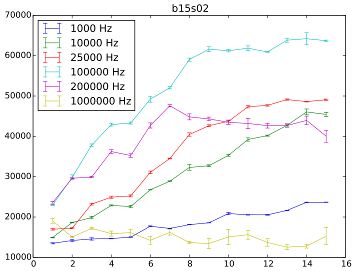


Figure 6: Material B15S02, length of compressed buffer vs. increasing number of input pins enabled.

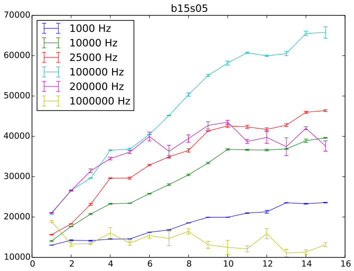


Figure 7: Material B15S05, length of compressed buffer vs. increasing number of input pins enabled.

compressed string rises sharply up to around 8-10 pins for all the materials. A threshold of saturation is further seen after this point. A possible explanation for the shape is that we add more pins, there are phase-offsets between input signals introduced due to imperfect scheduling on the Mecobo platform and potentially small interactions between the signals that give harder-to-compress output. As the number of pins increases, the amount of energy flowing through the system gives a more or less uniformly noisy measured signal. Manual inspection of the measured signal from the material confirms this.

Note that the 100KHz signal is higher than the rest. This is a combination of the measurement rate and bandwidth of the material. There is still relatively low damping seen on the signal at this rate, and 5 samples per cycle are sufficient to capture the main shape of the measured signal, and at the same time there is variation due to the frequency of output. At this frequency there is also ringing due to Gibbs phenomenon visible at high sample rates taken with an oscilloscope, giving rise to even more unpredictability.

The 1000Hz signal is almost undamped in the material, and it is therefore sharp and with a low amount of noise, making it easy to predict. The 1MHz signal is the exact opposite; very high damping ensures that the measured signal is mostly noise which should be relatively constant throughout the measurements, in particular at the sample rate we are limited to.

SPICE modelling

SPICE is a circuit simulation tool. By observing the behaviour of the material under certain inputs, it is possible to create a macroscopic 'pin-to-pin' model of a material slide. For each pin pair, we measure the DC resistance and voltage response under various frequencies, and observe the response via an oscilloscope or via the Mecobo platform. This enables us to produce models based on common circuit elements such as capacitors, inductors and resistors that mimic what we observe at this level. A practical use for SPICE models is often to tune circuits in such a way that one obtains so-called *impedance matching*, in which the impedance seen by the observing element matches its own internal impedance. This allows for maximum power in the transferred signal. In the context of evolution-in-materials, the model can be used for such a purpose as well; since the materials provide a broad range of 'loads', we can use a model of the macroscopic properties to tune our measurement circuitry in such a way as to increase our chances of

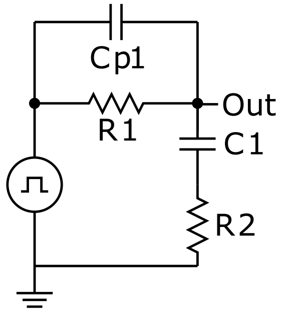


Figure 8: Simulation circuit, the output is measured from GND to the node between R1 and C1.

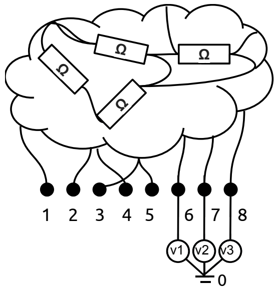


Figure 9: SPICE model used for the TSP experiments. Numbers correspond to SPICE nodes, v1,v2 and v3 are voltage sources. The cloud consists of randomly connected resistors of various values.

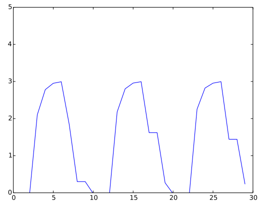


Figure 10: Mecobo capture of voltages for a 50KHz signal at 500KHz sample rate.

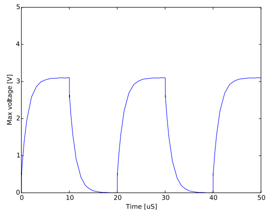


Figure 11: Spice output voltages for a 50KHz signal, simulated at 10ns timesteps.

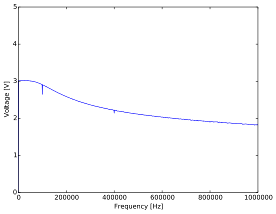


Figure 12: Mecobo capture of maximum voltage per frequency.

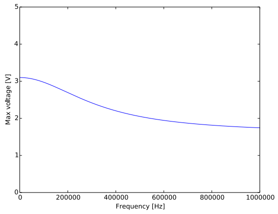


Figure 13: Spice output of maximum voltage per frequency.

finding emergent behaviours resulting from microscopic interactions in the material.

The model required to capture the effects of square waves seen at the level we are observing on can be simple. As discussed in the previous section, we believe that the main contributor to the filtering effects can be modelled by a single small filter, and this is reflected in Figure 8. By doing these measurements on all pin pairs and inserting one of these models between each pin pair, a model of the complete material slide can be constructed.

The model for this can be seen in Figure 8. This circuit is an RC low pass filter (R1 and C1) connected together with an RC high-pass filter (R2 and C2), along with a parasitic capacitance in parallel with the 'main' resistor (Cp1). The DC load of the system is captured by R1. The parasitic capacitance is added to model ringing, either due to Gibbs phenomenon (Hewitt and Hewitt, 1979) or simply parasitic capacitances and inductances resonating at their characteristic frequency). We observe these effects when applying certain frequencies to the material, which also can be barely seen in Figure 12 for the captured signal and Figure 10 for the model, albeit not to a large extent. In both cases a signal of 50KHz has been applied to the material. The signal output from the circuit is measured over C1 and R2, and the input is modelled by a pulse train voltage source. Looking at Figures 10 and 11, it is obvious that the signal captured with Mecobo (which used a sample rate of 500KHz), does not reproduce all the effects seen at simulation level, such as the ringing present due to Gibbs phenomenon and possibly other small parasitic inductances and voltages, both in the material and the measurement apparatus. We therefore used an oscilloscope to capture a more detailed version of the signal, in which more of the effects mentioned above can be observed. This begs a deeper question: are there events in the material occurring that we fail to capture with our measurements? Take inductance, as mentioned above for instance. Since there is current flowing through the material (the conductor), there must also be a magnetic field associated with this moving current. Isolated it is likely very small, but the sum of the fields could be measurable with Hall effect detectors (Ramsden, 2006), which could lead us to further improve the model by adding the correct inductances as to mirror the effects seen.

Travelling salesman

A further experiment was set up to attempt to solve the travelling salesman solved by Clegg in (Clegg et al., 2014) using a relatively simple SPICE model. In essence the problem as presented condenses into finding a way to configure the material into settling on a number voltages of different values. Each output pin is tagged with a city and the goal is to have the sorted values of the voltages on the output pins correspond to the shortest path between these cities, e.g. if pins 1, 2, 3 corresponds to the cities, and the shortest path between them is 2, 3, 1, the requirement is that $V(pin2) < V(pin3) < V(pin1)$. For details regarding the problem definition, we defer the reader to (Clegg et al., 2014).

Figure 9 shows our SPICE-model of this problem. We generate a 'cloud' consisting of nothing but resistors and connections (or 'nodes' in SPICE-terms) between them. An initial study of the problem revealed that we did not need to include capcitors to solve the problem. This is not surprising since a capacitor is an open switch with DC current. The first k SPICE-nodes (1 to 5 in the figure) are used as 'cities' and are only used as voltage measurement points. The remaining pins are connected to voltage sources whose voltages are adjusted with a (1+4) evolutionary strategy. The resistor-cloud-generating procedure uses an adjustable number of internal nodes and an adjustable number of generated resistors, giving us the ability to adjust the density of the cloud with these two parameters. In addition, the values of the resistors are drawn from a uniform distribution in a range also passed to the procedure, which enables tuning of the 'point-to-point' resistance of the network.

The instance we are seeking to provide a solution for is one of the presented solutions in (Clegg et al., 2014); a circular arrangement of 8 cities in which the shortest path is simply that– a circle. The cities are arranged on a grid, and the path length of the suggested solutions (as described above) is measured as a real number and taken as the fitness.

The vector being optimized is a 4-tuple consisting of 4 floating point values, (V_1, V_2, V_3, V_4), which maps directly to the voltages set on the voltage sources. Mutation is done by adding a number drawn from a standard normal distribution with $\mu = 0.0, \sigma = 2.0$ to one of the elements of the tuple. The SPICE simulation is run for 20ms using transient analysis with a resolution of 0.01ms. The final value of the measurements is the arithmetic average of the time series as measured across the nodes labeled as cities (i.e. nodes 1 to 8) and node 0 (which corresponds to ground).

We find that solutions are relatively quickly obtained (within 100 generations) in some of the randomly generated networks, while in others the search settles into a local optima and is unable to escape before the termination of the search which happens at 2000 generations.

This led us to investigating a few different parameter settings of the resistor cloud, the results of which can be seen in table 2. We generated 1000 networks for each parameter setting and counted the number of perfect solutions within 100 generations. As can be seen, there are relatively few solutions in all cases. Since we spent no significant time evaluating the different combinations, this is to be expected– the only conclusion that can be drawn from these numbers is that the composition of the resistor cloud matters when solving this type of problem. The somewhat unsurprising conclusion is that the composition of the materials used in the 'real' evolution in materials matters.

Nodes	**Resistors**	**Solutions**
35	150	30
35	300	19
35	500	32
60	150	69
60	300	11
60	500	34

Table 2: Results from running a search for a TSP solution in 1000 randomly generated resistor cloud networks.

Discussion and conclusions

How should we view the results in (Lykkebø et al., 2014), (Mohid et al., 2014) in which square waves are used as one of the configuration parameters then? The only observable property of the material pinpointed thus far is the fact that it will act as low pass/high pass signal filter. The different pin pairs thus act like 'frequency selectors' and it is possible to think of ways to do a form of computation with these (a frequency discriminator (Thompson, 1997) for instance), which could provide a mechanism that an evolutionary algorithm could potentially find and make use of. Another explanation that needs to be investigated further is the very real possibility that the fitness functions have been defined in such a way that they can use *noise* to solve the problem, and that lack of re-evaluations in the experiments lead to solutions that worked once, but only by chance; by encoding the solution to the problem in the input and simply evolving a way to reconstruct the solution from the input and the noise added by the material.

As for configuration of the material using static DC voltages, we have shown that the material in this case behaves mostly as a resistor network. As we have shown in our simple SPICE model shown in Section IV it is indeed possible to solve the problem instance solved by Clegg in (Clegg et al., 2014) using a network of resistors. One important point to note is that the material is rich and more flexible than one single purpose-built resistor network; it contains a large amount of various such networks, each of which can be exploited by evolution to solve a number of problems.

An issue regarding this way of formulating the problem is that we are limited by the resolution of the measurements. A common ADC such as the one used in the Mecobo daughterboard, has a resolution of 12 bits, giving a maximum of 2^{12} different voltages, effectively limiting the number of cities to 4096. This is the best case, often times one can only hope to achieve 10-11 bits of resolution, depending on how well-matched the impedance of the load is to the measurement apparatus, which again is affected by the characteristics of the material which can vary by large amounts, as we have seen previously in this paper. One can of course invest in even higher-resolution measurement apparatus to achieve well over 20 bit resolution but the point of diminishing returns in terms of practicality of building a computing system seems likely to be reached quite fast by making further advancements in this.

For the results in (Kotsialos et al., 2014) we suggest that it might be hard to reproduce these results as well. Making a XOR gate out of a resistor network is not possible since a way of inverting a signal is needed; but it is not unlikely that it is possible to achieve with a noisy system, as there are an abundant amount of transistors, diodes and let's not forget, *environmental* noise readily available in a physical system. This of course leads us into the question of where to draw the line between the computational entity, the measurement apparatus and the input. For more discussion around this, see (Lykkebø et al., 2014).

As such, we conclude that future work using the SWCNTs thus needs to take particular care with proving that it is 1) better than a random search and 2) that the results obtained are repeatable with a high degree of confidence. One way of achieving this would be to operate *within the bandwidth* of the material, such that we can be sure we are measuring actual signal response and not noise. The second, obvious way, is to always discard unstable solutions if they fail to reproduce their behaviour a number of times. A third point to note is that we must construct the fitness functions in such a way as to minimize the evolutionary process' natural tendency to exploit unwanted effects such as noise.

Much work remains before we can draw final conclusions regarding the computational properties of random carbon nanotubes. One dimension that is immediately interesting in terms of richness is movability of the material, since geometric properties play an important role in solid state physics. Introducing a more viscous environment for the CNTs to move around in could prove fruitful, since there is evidence that the tubes are capable of self-organizing (Belkin et al., 2015), and that geometry matters (Ebbesen et al., 1996) and just as a human designer is free to exploit the spatial properties of electro-material interactions, so is it possible for an evolutionary process to do the same.

Acknowledgements

The research leading to these results has received funding from the [European Community's] Seventh Framework Programme ([FP7/2007-2013] [FP7/2007-2011]) under grant agreement no [317662].

References

Ashby, W. R. (1956). *An introduction to cybernetics*. Chapman & Hall, London.

Belkin, A., Hubler, A., and Bezryadin, A. (2015). Self-assembled wiggling nano-structures and the principle of maximum entropy production. *Sci. Rep.*, 5.

Broersma, H., Gomez, F., Miller, J. F., Petty, M., and Tufte, G. (2012). Nascence project: Nanoscale engineering for novel computation using evolution. *International Journal of Unconventional Computing*, 8(4):313–317.

Cariani, P. (1993). To evolve an ear: epistemological implications of gordon pask's electrochemical devices. *System Research*, 10(3):19–33.

Clegg, K. D., Miller, J. F., Massey, K., and Petty, M. (2014). Travelling salesman problem solved 'in materio' by evolved carbon nanotube device. In Bartz-Beielstein, T., Branke, J., Filipič, B., and Smith, J., editors, *Parallel Problem Solving from Nature – PPSN XIII*, volume 8672 of *Lecture Notes in Computer Science*, pages 692–701. Springer International Publishing.

Ebbesen, T. W., Lezec, H. J., Hiura, H., Bennett, J. W., Ghaemi, H. F., and Thio, T. (1996). Electrical conductivity of individual carbon nanotubes. *Nature*, 382(6586):54–56.

Harding, S. L. and Miller, J. F. (2004). A tone discriminator in liquid crystal. In *Congress on Evolutionary Computation(CEC2004)*, pages 1800–1807. IEEE.

Harding, S. L. and Miller, J. F. (2007). Evolution in materio: Computing with liquid crystal. *Journal of Unconventional Computing*, 3(4):243–257.

Harding, S. L., Miller, J. F., and Rietman, E. (2008). Evolution in materio: Exploiting the physics of materials for computing. *Journal of Unconventional Computing*, 3:155–194.

Hewitt, E. and Hewitt, R. (1979). The gibbs-wilbraham phenomenon: An episode in fourier analysis. *Archive for History of Exact Sciences*, 21(2):129–160.

Kolmogorov, A. N. (1965). Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1:1–7.

Kotsialos, A., Massey, M., Qaiser, F., Zeze, D., Pearson, C., and Petty, M. (2014). Logic gate and circuit training on randomly dispersed carbon nanotubes. *International journal of unconventional computing.*, 10(5-6):473–497.

Lykkebø, O. R., Harding, S., Tufte, G., and Miller, J. F. (2014). Mecobo: A hardware and software platform for in materio evolution. In Ibarra, O. H., Kari, L., and Kopecki, S., editors, *Unconventional Computation and Natural Computation*, Lecture Notes in Computer Science, pages 267–279. Springer International Publishing.

Lykkebø, O. R. and Tufte, G. (2014). Comparison and evaluation of signal representations for a carbon nanotube compuational device. In *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pages 54–60.

Massey, M. K., Pearson, C., Zeze, D. A., Mendis, B. G., and Petty, M. C. (2011). The electrical and optical properties of oriented langmuir-blodgett films of single-walled carbon nanotubes. *Carbon*, 49:2424.

Miller, J. F. and Downing, K. (2002). Evolution in materio: Looking beyond the silicon box. In *2002 NASA/DOD Conference on Evolvable Hardware*, pages 167–176. IEEE Computer Society Press.

Miller, J. F., Harding, S., and Tufte, G. (2014). Evolution-in-materio: evolving computation in materials. *Evolutionary Intelligence*, 7(1):49–67.

Mohid, M., Miller, J. F., Harding, S. L., Tufte, G., Lykkebo, O. R., Massey, M. K., and Petty, M. C. (2014). Evolution-in-materio: Solving function optimization problems using materials. In *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pages 1–8.

Nichele, S., Laketic, D., Lykkebø, O. R., and Tufte, G. (2015). Is there chaos in blobs of carbon nanotubes used to perform computation? In *7th International Conference on Future Comp. Tech. and Applications, IN PRESS*. XPS Press.

Nichele, S. and Tufte, G. (2013). Measuring phenotypic structural complexity of artificial cellular organisms - approximation of kolmogorov complexity with lempel-ziv compression. In *Innovations in Bio-inspired Computing and Applications - Proceedings of IBICA 2013, August 22 -24, 2013 - Ostrava, Czech Republic*.

Pask, G. (1959). Physical analogues to growth of a concept. *Mechanisation of Thought Processes*, pages 877–922.

Ramsden, E. (2006). *Hall-Effect Sensors - Theory and Application (2nd Edition)*. Elsevier.

Thompson, A. (1997). An evolved circuit, intrinsic in silicon, entwined with physics. In *1st International Conference on Evolvable Systems (ICES96)*, Lecture Notes in Computer Science, pages 390–405. Springer.

Apparent actions and apparent goal-directedness

Martin Biehl and Daniel Polani

University of Hertfordshire, Hatfield, UK

m.biehl@herts.ac.uk

Introduction In human history countless phenomena have been (wrongly) attributed to *agents*. For instance, now science believes there are no gods (agents) of lightning, thunder and wind behind the associated phenomena.

In physics (assuming quantum decoherence) the universe is modelled as a state space with a dynamical law that determines everything that happens within it. This however, is incompatible with most notions of agency (cf. Barandiaran et al., 2009) which require *actions*: For an agent candidate to have actions it must be able to "make something happen" as opposed to only "have things happen to it".

Here we ask which single sequences of partial observations may *appear* to contain agency to a *passive observer* who has its own memory. For this we define measures of *apparent actions* and *apparent goal-directedness*. Goal-directedness is another feature commonly attributed to agents. We here ignore whatever causes the appearances and the concept of individuality of agents.

Apparent actions We assume that a passive observer perceives a sequence S of sensor values $(s_1, ..., s_T)$ with $T \in \mathbb{N}^+$. At each instance t of the sequence, the observer has a memory or knowledge state m_t which gives us a second sequence $M = (m_1, ..., m_T)$. The memory state of the agent might contain models of the observations or not. In the course of a sequence S there is an *apparent action for observer* M if for $r, t \in \{1, ..., T\}$ we have $(s_r, m_r) = (s_t, m_t)$ and $(s_{r+1}, m_{r+1}) \neq (s_{t+1}, m_{t+1})$.

The intuition is that, since the same observation $s_r = s_t$ at different times r, t is followed by different observations and the observer's states $m_r = m_t$ do not indicate / predict the difference, the observer suspects a hidden mechanism causing the difference. We assume that the observer interprets all signs of hidden mechanisms as (apparent) actions.

Using the empirical distribution $p_{S,M}(s', s, m) = \frac{1}{T}\sum_{t=0}^{T} \delta_{s's_{t+1}}\delta_{ss_t}\delta_{mm_t}$ (δ_{xy} is Kronecker's delta) we can quantify the extent of apparent actions along the sequences as the conditional entropy $\mathrm{H}(S'|S, M)$.

Apparent goal-directedness Our observer attributes the complete sequence S to be the result of an agent's strategy. Note that even if there is no apparent action along a subsequence this could be due to the agent trying to avoid detection. The idea is that any directedness reveals itself as some pattern within the sequence and any pattern in the sequence will increase the compressibility of the sequence. So we here define *apparent goal-directedness of the observed sequence* S as its compressibility. Using a common compression algorithm (e.g. gzip) Z we can estimate compressibility as $l(S) - Z(S)$ where $l(S)$ is the binary length of all the data in the observed sequence and $Z(S)$ the binary length of the compressed data. Note that an adversary's goal-directedness can remain undetected only if S is completely random.

Examples Requiring both apparent action and apparent goal-directedness leads to the following classifications: 1.) A Brownian particle exhibits apparent actions but very low apparent goal-directedness. 2.) A ball thrown through the air exhibits no apparent actions if $m_t = s_{t-1}$ i.e. if memory contains the previous observations. Together the s_{t-1} and s_t are enough to get linear momentum and position of the ball which together determine its trajectory. Apparent goal-directedness is high as the equations of motion compress the flight path. 3.) A thief trying to guess a safe combination exhibits apparent actions as every time a new combination is tried out it starts in the same initial position. It also exhibits some goal-directedness, as it never tries the same combination twice, which is a pattern.

Conclusion The apparent notions identify agency in example systems and take into account capabilities of the observer. To fool the observer a (visible) adversary has to *a*) be predictable to the observer (no apparent action) or *b*) rely on randomness (luck) to achieve its goal (no goal-directedness).

References

Barandiaran, X. E., Paolo, E. D., and Rohde, M. (2009). Defining agency: Individuality, normativity, asymmetry, and spatio-temporality in action. *Adaptive Behavior*, 17(5):367–386.

Evolutionary Progress in Heterogenous Cellular Automata (HetCA)

David Medernach[1], Jeannie Fitzgerald[1], Simon Carrignon[2], and Conor Ryan[1]
[1]Biocomputing and Developmental Systems Group, Department of Computer Science & Information Systems,
University of Limerick, Ireland
[2]Computer Application in Science and Engineering (CASE), Barcelona Supercomputing Center, Barcelona, Spain
david.medernach@ul.ie, jeannie.Fitzgerald@ul.ie, simon.carrignon@bsc.es, conor.ryan@ul.ie

Abstract

Although very controversial in the field of evolutionary biology, the notion of *evolutionary progress* is nevertheless generally accepted in the field of *Artificial Life*. In this article we adopt the definition proposed by Shanahan (2012) to study the existence of evolutionary progress in an evolutionary simulation which we call HetCA. HetCA is a *heterogeneous cellular automata* characterized by its ability to generate open ended long-term evolution. In this study, we measure evolutionary progress on three criteria: the robustness, size and density of generated genotypes. Our results demonstrate that the oldest genotypes in terms of evolutionary time are frequently the most robust, and that phenotypic density is higher for genotypes collected later in the evolutionary process.

Introduction

Evolutionary progress (EP) hypothesizes that evolution tends toward a goal such as greater complexity of individuals. It's a truism to speak about EP in terms of Evolutionary Computation (EC) where evolution is used as an optimisation method to solve problems. But the existence of EP is much less obvious in biology which evolves in an open ended evolutionary process. HetCA, Medernach et al. (2013), is a heterogenous cellular automata (CA) evolutionary simulation which represents a version of *open-ended evolution*.

In this paper, we measure evolutionary progress in HetCA using three traits: the density, size and robustness of evolved individuals. To do so we compare the properties of genotypes taken at different stages of the evolutionary process.

The study of these three features allows us to analyze more precisely the nature of evolutionary strategies in HetCA. We hypothesize that size of the genotype together with robustness and density of its phenotype may serve as useful measures of the effectiveness of that genotype. We examine the existence of a correlation between these measures.

Our paper is organized as follows: firstly, in the Background section, we review EP and CA. Then, in the Methodology section, we present experimental settings and define the measures which we have chosen to use as evidence for the existence of EP. Following this, we present the simulation results in the Results section and finally, in the Discussion section, we discuss those results and explore some specific examples to discuss their behaviour qualitatively.

Background

Evolutionary Progress

The popular belief that a form of evolutionary progress emerges naturally as a result of the process of natural selection may seem obvious, but as described by Shanahan (2000), it is a controversial view in the field of biology. While the notion has several supporters, including Richard Dawkins and Ernst Mayr, most evolutionary biologists, like for example, William Provine and Stephen Jay Gould strongly criticise this idea: “Progress is a noxious, culturally embedded, untestable, nonoperational, intractable idea that must be replaced if we wish to understand the patterns of history”, Gould (1988).

There are various reasons for this rejection of the EP concept in biology. For one thing, the notion of EP is sometimes regarded as a remnant of a very anthropocentric view of the classification living beings (a legacy of Aristotle), which can also be seen in several predecessor of the Darwinian model of natural selection, such as theories proposed by Buffon or Lamarck.

Additionally, the concept of EP is usually linked to the existence of an increase in the complexity of a living being. And this idea is challenged in evolutionary biology for various reasons, summarized in McShea (1991).

Another argument, which could be seen to weaken the EP hypothesis, has been put forward by researchers such as Johnson et al. (2012) in which they highlight numerous examples of species losing previously acquired evolutionary traits, possibly induced by ecological changes.

On the other hand, in the field of *Artificial Life*, though the creation of artificial long-term EP in open ended simulation has not received consensus, the existence of evolutionary progress in the living is often taken for granted as illustrated by, for example, Ciliberti et al. (1999) and its association with the evolution of complexity is presented as one of the goals of *Artificial Life* by Bedau (2003).

This dissemblance in approach may be explained by the proximity of the field of *Artificial Life* with other evolutionary paradigms of optimization and problem solving which are similarly inspired by natural selection, such as Genetic Programming and Genetic Algorithms, where the existence of evolutionary progress is not disputed.

But the reason could simply be that the concept of evolutionary progress has not yet been fully formulated and developed. In that regard, Shanahan (2012) proposed a definition of evolutionary progress[1] as "intergenerational directional change embodying improvement in the properties characterising a population of biological entities". A change should be considered directional if it "has a direction over a given time interval" and therefore "if the value of one of its properties increases during that time interval". It is important to note that, according to this definition, evolution acts on a multitude of populations themselves having multiple characteristics. It is, then, quite possible that evolutionary progress exists for a particular lineage of living beings while, in another lineages, for the same trait, there is no evolutionary progress or its direction is reversed.

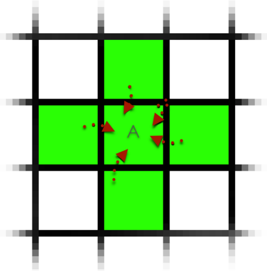
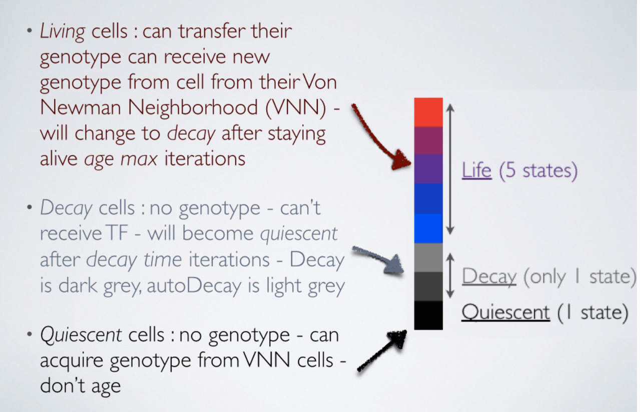


Figure 1: **Genotype transfer** at the start of a CA iteration, if cell A is not in Decay it will randomly receive a genotype from any cell shown here in green (Von Neumann (VN) neighboring cell A and cell A itself) that is neither in Decay nor Quiescent.

Evolutionary computation and open ended simulations

One of the problems facing researchers who use natural selection as an optimization mechanism in EC, is a decrease in the diversity of individuals during evolution which may lead to premature convergence to a local optimum Hornby (2006). Several biologically inspired solutions for avoiding premature convergence have been successfully tested. For example, inspired by an evolutionary arms race, Hillis (1990) used the coevolution of two populations[2] and Lessin et al. (2013) employed an intermediate step goal to generate behavioral complexity in evolved virtual creatures. In both of these studies the researchers examined evolution that was goal orientated but also in some sense *open ended*. Also, in Evolutionary robotics, Haasdijk et al. (2014), among others studied the ability to combine, in a single simulation, survival of the organism together with resolution of tasks. Thus, it may be useful to analyze the mechanisms that avoid or ameliorate premature convergence when evolution is open ended.



- *Living* cells : can transfer their genotype can receive new genotype from cell from their Von Newman Neighborhood (VNN) - will change to *decay* after staying alive *age max* iterations
- *Decay* cells : no genotype - can't receive TF - will become *quiescent* after *decay time* iterations - Decay is dark grey, autoDecay is light grey
- *Quiescent* cells : no genotype - can acquire genotype from VNN cells - don't age

Figure 2: **Specificities of states in HetCA**. Here $agemax =$ 7 iterations and $decay\,time = 375\,to\,1875$ iterations as specified in Table 1. Age of *living cells* increase each generation as long as it stay alive and is reboot to zero if the cell become *Quiescent* or *Decay*.

To the best of our knowledge, while CA models are widely used in *Artificial Life* for such things as artificial chemistry simulation Cole and Muthukrishna (2014) or to model various evolutionary processes Wolfram (2002) no attempt has yet been made to directly investigate EP in heterogeneous cellular automata. HetCA is an *Artificial Life* model designed to achieve some form of open-ended evolution[3]. Its purpose is similar to simulations such as Tierra Ray (1993) or Avida Adami et al. (1995), systems where a competition between replicating computer programs occurs in a virtual machine.

As described by Medernach et al. (2013), HetCA is designed as a heterogenous cellular automata, using several cell categories: Living cells, decay cells and quiescent cells. Every living cell has, in addition to its state, a genotype that determines its transition rules. These genotypes mutate and may spread to neighboring cells as shown in Figure 1. Moreover, the application of mechanisms such as Aging, Quiescence and Decay as shown in Figure 2, demonstrate that even a minimal survival strategy of genotypes implies

[1] This definition no longer use to the notion of complexity which itself is the subject of debate that we will not have room to discuss here.

[2] A population of solutions of sorting networks and a population of problems are co-evolved.

[3] Evolutionary process in which novel artifacts are continuously produced.

some form of cooperation between some cells[4] as illustrated in Figure 3. The HetCA model achieves long term dynamics, some form of *open-ended evolution* and different *ecosystems* characterized by distinctive patterns.

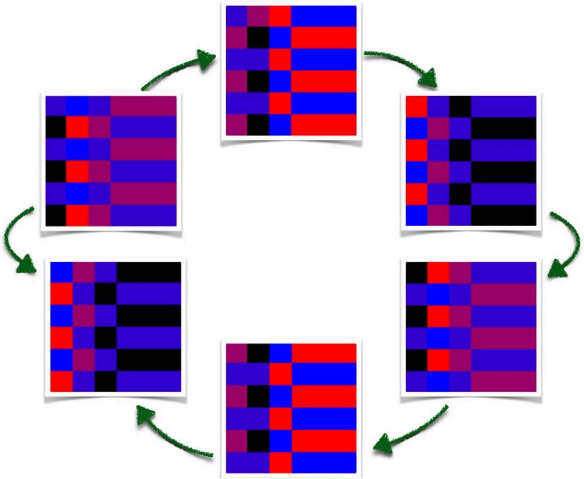


Figure 3: **Six steps survival strategy** from one possible phenotype of a genotype extracted at iteration 300000 in HetCA, tested here in a randomly initialized homogenous CA. This phenotype does not provide cells with the opportunity to grow old enough to decay before an evolutionary step changes them into quiescent cells. In doing so the cells lose their own genotype to facilitate the survival of the genotypes of neighboring cells. The meaning of the colour coding is described in Figure 2.

Methodology

In this section we analyze three genotypic properties as potential indicators of evolutionary progress: robustness, size and density.

Collection of genotypes

In order to assess the existence of EP we created a collection of genotypes at various stages of the evolutionary process, in the following manner: We performed 30 simulations, each on 500000 iterations with the parameters[5] listed in Table 1. The possible genotypes of an individual are its transition rules encoded with CA-LGP using the function set depicted in Table 2. Mutation of genotypes is enabled and we use the Micro/Marco-mutation of CA-LGP described in Medernach et al. (2013). For each simulation we saved the *most common genotype* (most frequently occurring) in iterations 5, 1000, 5000, 50000, 300000 and 500000. We have chosen to use iterations 5, 1000, 5000 and 50000 as they correspond to four distinct stages of the typical evolutionary process in HetCA and are characterized by four very differents environments as shown in Figure 4. Iteration 500000 was chosen because it is the final iteration of studied simulations, and iteration 300000 was selected as an intermediate step between iteration 50000 and iteration 500000.

The choice of the most common genotype may seem arbitrary, but it was not realistic to process all genotypes at each iteration of the simulation whereas the frequency is a naive, but nonetheless efficient criterion for assessing its representativeness and its success at any stage of the simulation.

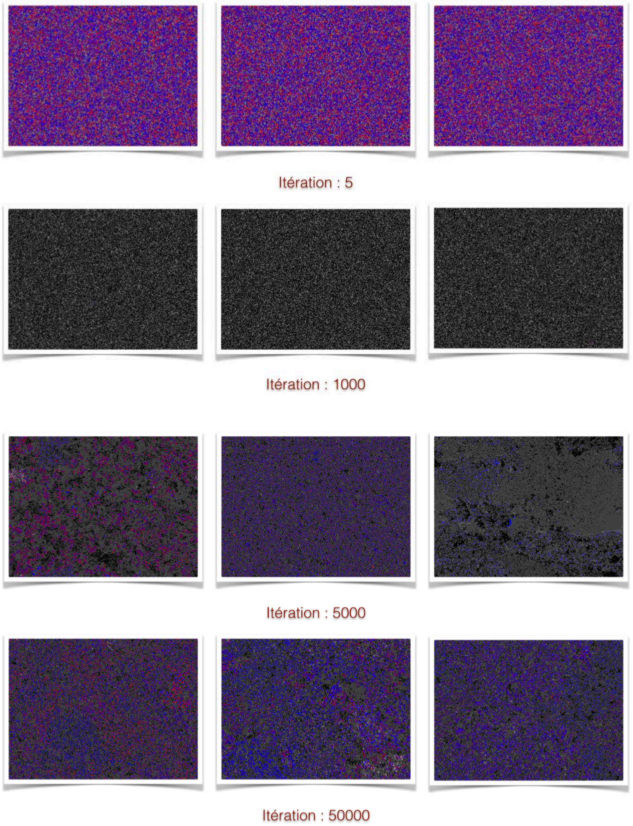


Figure 4: **Typical HetCA grid** at iterations 5, 1000, 5000 and 5000. Living cells are very frequent at iteration 5; at iteration 1000 most cells are in decay or in a quiescent state; and from there the population of living cells increases until iteration 50000. Color coding is described in Figure 2.

Evaluation of genotype robustness

To assess the robustness of the individual we measure its ability to survive in different environments. One could compare this measurement with the definition of evolutionary progress proposed in Dawkins (1997): “a tendency for lineages to improve cumulatively their adaptive fit to their particular way of life, by increasing the numbers of features

[4] Copying its genotype into a nearby quiescent and then committing “cellular suicide”, changing to quiescent state and therefore rebooting its life counter.

[5] Parameters are identical to HetCA-a7 in Medernach et al. (2013).

Parameter	Value
Number of Living states	5
Successive living iterations before decay	7
Number of iterations for decay	375-1875
Direct transition to decay	enabled
Size of the grid	400x300
Grid boundaries	toric grid
Transition Rule (TR)	CA-LGP
Maximum (TR) size	50 program statements
Genotype copy neighboring	Von Neumann
Transition rule neighboring	Moore

Table 1: **HetCA parameters**.

op. name	**action** on inputs (x, y)
abs	$\|x\|$
plus	$x + y$
delta	1, if $\|x - y\| < 1/10000$; 0 o.w.
dist	$\|x - y\|$
inv	$1 - x$
inv2	safeDiv$(1, x)$
magPlus	$\|x + y\|$
max	$\max\{x, y\}$
min	$\min\{x, y\}$
safeDiv	x/y if $\|y\| > 1/10000$; 1 o.w.
safePow	x^y, if defined; 1 o.w.
thresh	1, if $x > y$; 0 o.w.
times	xy
zero	1, if $\|x\| < 1/10000$; 0 o.w.

Table 2: **Function set**.

which combine together in adaptive complexes".

The robustness of an individual is measured by comparing genotypes in a pairwise fashion in series of simulations where we disable mutations. We compare each genotype to every other genotype saved from a different run after a *different* number of iterations. We don't compare genotypes from the same runs because they have already competed in their evolutionary history, and making these comparisons could potentially skew results. Nor do we compare two genotypes collected after the same number of iterations of the cellular automata because this would not provide any information about EP.

Half of the cells, randomly selected, are initialized with the first genotype, the other half with the second one. All cells are initialized at a random living state. The simulation is stopped either after 50000 iterations[6], as illustrated in Figure 5, or when more than 99% of the living cells share the same genotype. After the simulation is stopped, the most frequent genotype is considered to be more robust than the other one due to it's dominance of the environment. To conduct this experiment in a reasonable amount of time we choose to use only genotypes from the first 10 simulations at the 6 previously chosen iterations, which represents $6 \times 10 = 60$ genotypes and $45 \times 60/2 = 1350$ simulations.

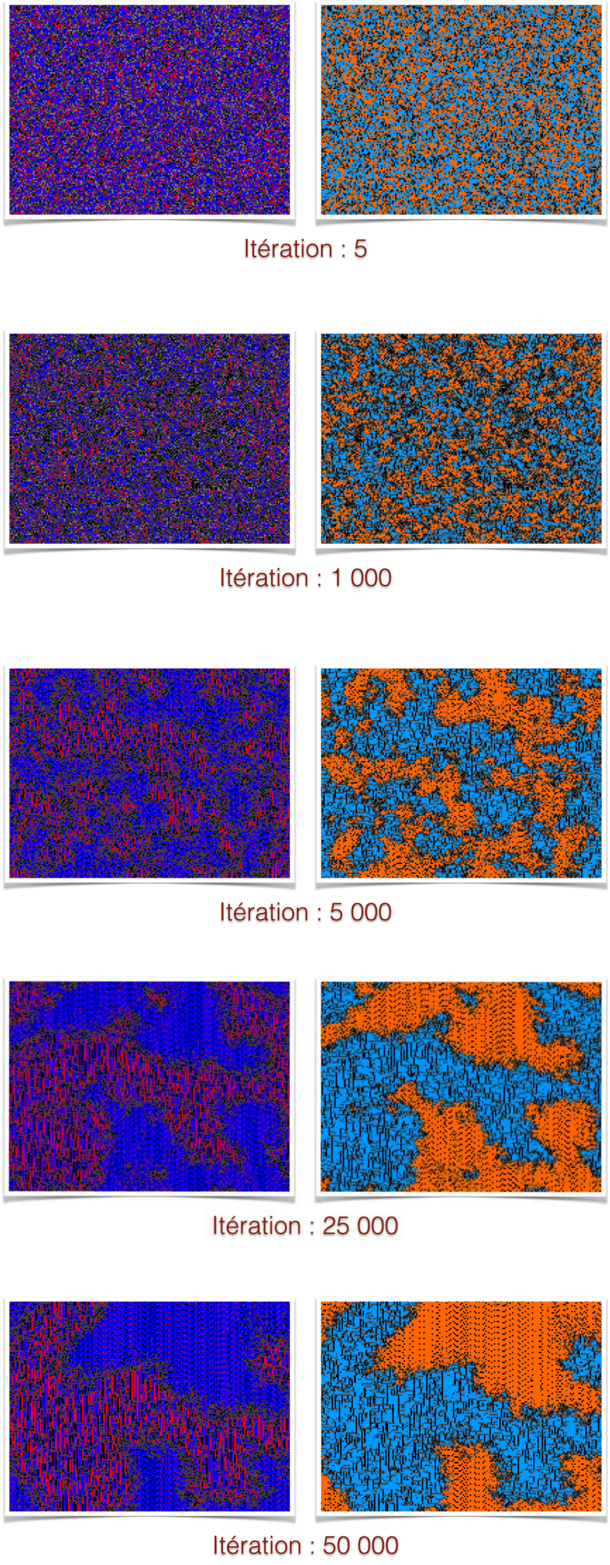


Figure 5: **Formation of cells clusters in a robustness test**. On the right column the cells are depicted with their current states (color coding is described in Figure 2), on the left column the repartition of the two genotypes is depicted. Cells that don't curently have a genotype because they are in a decay or quiescent state, are represented with their current states on both sides.

[6] Long simulations are computationally expensive , the limit of 50000 iterations was chosen after informal tests indicating that this one was rarely exceeded and in cases where it was, the trend, in terms of majority genotype, was never reversed.

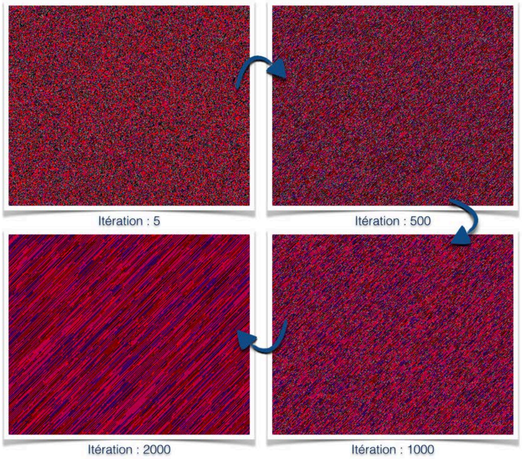


Figure 6: **Density test**. At each iteration the density rho_i is measured as the proportion of living cells. It is the proportion of living cells among all the cells. (color coding is described in Figure 2)

Evaluation of phenotype densities

The density of each genotype is evaluated by a simulation where mutations are disabled and each cell is initialized at a random living state with the tested genotype as transition rule, as illustrated in Figure 6. The simulation is run for 2000 iterations. The density measure ρ of a genotype is the average number of living cells during the simulation:

$$\rho = \frac{\sum_{i=1}^{2000}(\rho_i)}{2000} \times 100 \quad and \quad \rho_i = \frac{n_{alive}}{S}$$

Where ρ_i is the density of the phenotype[7] for the iteration i, n_{alive} is the number of cells which current state is one of the alive states[8] and S is the size of the grid of the cellular automata[9]. The phenotypic densities of all the $30 \times 6 = 180$ genotypes are computed.

Evaluation of genotype sizes

In evolutionary biology, according to Lynch and Conery (2003) the increase of size of individual genotype is caused by genetic drift and linked to population size (N_e)[10]. Similarly, in evolutionary computation size is frequently studied, if only because of the potentially high computational cost associated and a potential correlation with overfitting Fitzgerald and Ryan (2014). More specifically, in evolutionary algorithms (EA), different studies show that size is not correlated with the ability of individuals to solve the presented task. We use the number of program statements (n_{prog}) as a measure of the genotype size. Sizes of all the $30 \times 6 = 180$ genotypes are computed.

Note that two of the three studied traits[11] are bounded and directly measurable while robustness is a relative criterion.

Results

Robustness

In accordance with the hypothesis proposing the existence of evolutionary progress, Table 3 shows that during the pairwise comparison of the genotypes collected at different stages of the simulation, the oldest genotypes are frequently the most robust. The scores are higher than 89% in eleven out of fifteen cases, and even the smallest margin of 59% reported for the comparison of genotypes collected after 5000 and 1000 iterations, is the only non-significant using the binomial test at p = 0.05. Table 4 shows that not counting the tests that reach 50000 iterations slightly increases the win rate of the oldest genotypes. Not surprisingly, Table 5 indicates that robustness increases with increasing number of iterations. The inclusion or non-inclusion of simulations where no genotype had reached 99% dominance of living cells does not significantly impact the results, this shows that 50000 iterations are sufficient to perform this test.

	5	**1000**	**5000**	**50000**	**300000**	**500000**
5	-	347	339	375	34	384
1000	87%±7	-	11397	2586	2172	2465
5000	98%±2	59%±10	-	3962	3468	5203
50000	96%±4	97%±3	96%±4	-	8890	7612
300000	100%	100%	97%±3	68%±10	-	11224
500000	98%±2	95%±4	93%±5	76%±9	66%±10	-

Table 3: **Pairwise comparison of the robustness of genotypes collected at different stages of evolution**: The robustness of genotypes is shown under the diagonal of the table, whereas average final iterations are shown above the diagonal. The robustness is the proportion of the comparative tests where the row genotype was more prevalent than the column genotype.

Size

Results in Figure 7 do not show significant differences, between the size of the genotypes collected in iterations 1000, 5000 and 50000 using the Welsh test at p = 0.05. The genotypes collected at iterations 5 are significantly smaller while those taken at iterations 300000 and 500000 are significantly larger. The average size of the genotypes, selected in iterations 300000, is very close to the maximum of 50 and does not significantly increase at iteration 500000.

[7] We call here phenotype, the pattern drawn by the states of cells sharing the same genotype.

[8] Alive states are all the states that are neither Quiescent state nor Decay state.

[9] $300 \times 400 = 120000$ cells in those simulations.

[10] An increase in the size of the genotype corresponding to a reduction of the size of the population.

[11] Size of the genotype and density of its phenotype.

	5	1000	5000	50000	300000	500000
5	-	340	333	368	34	381
1000	87%±7	-	6015	2055	1646	2471
5000	98%±2	63%±10	-	2880	2411	3676
50000	96%±4	97%±3	96%±4	-	6955	6649
300000	100%	100%	98%±2	70%±10	-	7441
500000	98%±2	95%±4	94%±5	78%±9	67%±10	-

Table 4: **Pairwise comparison of the robustness of genotypes collected at different stages of evolution without tests reaching 50000 iterations**: The fields under the diagonal detail robustness of genotypes whereas, those on top of the table diagonal show the average final iteration. The robustness is the proportion of the comparative tests where the row genotype was more prevalent than the column genotype. Whereas Figure 3 simulations that reached 50000 iterations are not included.

Age	Robustness	Ending iteration
5	4%	295
1000	27%	3793
5000	34%	4873
50000	70%	4685
300000	80%	5157
500000	87%	5377

Table 5: **Robustness of genotypes collected at different stages of evolution**: Age is the number of iterations of the cellular automata after which collection of genotypes took place. The robustness is the proportion of the comparative tests where the tested genotype was the most prevalent. The ending iteration is the average number of iterations before the comparative tests terminated.

Density

Figure 8 shows that phenotypic density is higher for genotypes collected later in the evolutionary process. However there was no significant difference, using the Welsh test at p = 0.05, between iterations 1000 and 5000. Even if it is significant the difference in density, 47% against 48%, is very low between iterations 300000 and 500000 even thougth this is the second longest interval. It should be noted that, for genotypes extracted at iterations 1000, live cells have been completely extinguished before the simulation reaches 2000 iterations[12]. The average density of these same genotypes before extinction is 46% which would rank their density between those extracted at iterations 50000 and iterations 300000.

Discussion

The measures of evolutionary progress that we use here are pretty tough compared to the definition proposed by Shanahan (2012). Not only are the different genotypes collected within the same simulation not necessarily generated from a single evolutionary lineage. But we also collect individuals from different simulations and thus from independent evolutionary processes.

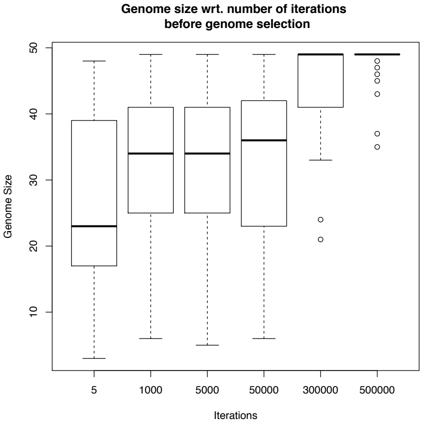


Figure 7: **Size of genotypes collected at different stages of evolution**: Iterations is the number of iterations of the cellular automata that occurred before the collection of the genotype, The size is expressed as the number of program statements (n_{prog}) used in genotypes.

Nevertheless, at first sight we detect evolutionary progress in HetCA, between iteration 5 and 500000. There are several periods of stasis[13] during this process but the direction of progress is never reverses and therefore changes in the three traits studied here are directional. Weak or nonsignificant differences between the traits analyzed in iterations 1000 and 5000 could be explained by the fact that there are relatively few evolutionary steps between them. However, a significant difference exists between the genotypes selected at iteration 5 and those selected at iteration 1000 although the number of iterations between them is only 995 iterations. This is likely due to the differences between the typical environment in iterations 5 and 1000 as depicted in Figure 4. At iteration 5, the critical part of the selection is the ability to survive and reproduce as quickly as possible in the “primordial soup” of the early iterations of HetCA, whereas at iteration 1000 the greater part of the cells are in decay and selection is made on the ability of a small group of cells to survive without saturating a reduced space.

The increase in cell density seems quite logical. The density of the phenotype being an obvious evolutionary advantage in HetCA. It is interesting to note that while there maybe periods of stasis in the increase of density, between iterations 1000 and 5000 the values for robustness continue to increase. This demonstrates that possible evolutionary strategies are not limited to density increase.

However, if the three measures used here show directional changes, small modifications of the experimental protocol

[12] It has also occurred for two genotypes extracted at iteration 1000 and one extracted at iteration 5000.

[13] For robustness and density.

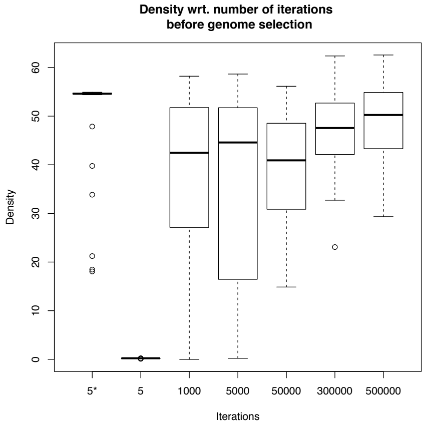


Figure 8: **Density of genotypes collected at different stages of evolution**. 5* is the density computed at iteration 5 before extinction of living cells.

could reverse this trend. Especially if the density measurement was stopped before the extinction of genotypes collected at iterations 5, in which case, they would have had a higher density than genotype collected at iteration 50000 as is shown in Figure 8. It is also quite difficult to assess whether the study period, 500000 iterations is sufficient to observe potential change in trend, and the EP seems to be slowing down in the last 200000 iterations.

The fast increase of genotype size between iteration 5 and 1000 could be the result of differences in size between random genotypes with which we initialize the simulation, those with viable strategies being maybe longer on average than others. We tested this hypothesis in the Table 6, it is checked but does not seem sufficient to explain this difference alone. An additional explication could be genetic drift, because as illustrated in Figure 4 the population, N_e, is very small during this interval.

	Average Size	Proportion
Effective survival strategy	29	0.5%
Ineffective survival strategy	25	99.5%

Table 6: **Survival strategy and size**: We randomly generated genotypes then tested, one by one, their ability to survive on 40×30 size grids (in simulations without mutations and where all cells are initialized with the same genotype). Genotypes extinct before iteration 5 have not been taken into account; Genotypes extinct before iteration 100 are considered as having ineffective strategy; genotypes still having living cells at iteration 100 are considered using an effective strategy. We performed this test on 100000 genotypes.

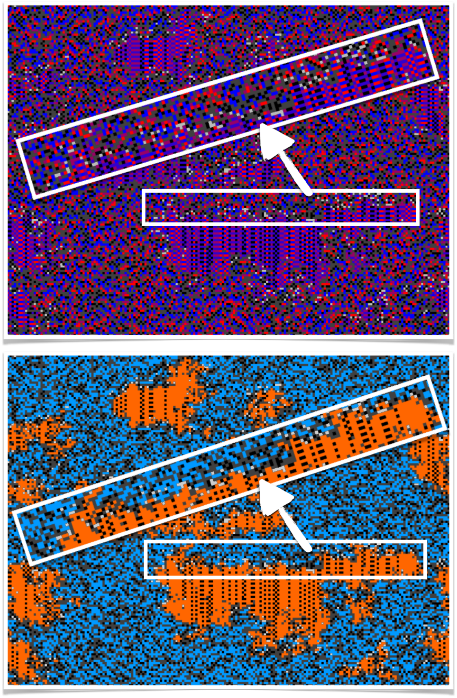


Figure 9: **Altruistic decay:** The light gray cells are cells in auto-decay, they appear here in areas contested by the two genotypes, helping to create a barrier between them. On the top image, cells are depicted with their current states (color coding is described in Figure 2), on the bottom image the repartition of the two genotypes is depicted. Cells that don't have a genotype (in decay or quiescent state) are represented with their current states on both sides.

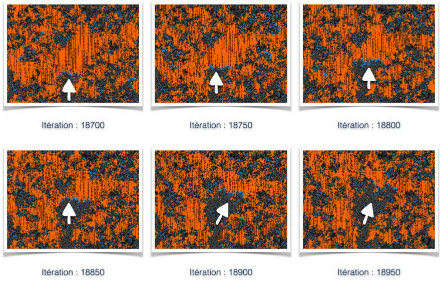


Figure 10: **Evolutionary strategy**: In these images we can see that cells with blue genotype, despite their low density, are effectively able to eliminate cells with the orange genotype when they are in contact.

Qualitative analysis

The pairwise comparison of the genotypes collected also facilitates some qualitative remarks on the nature of the interactions observed in HetCA. As can it be seen in Figure 5, groups of cells sharing the same genotype are formed very quickly. This reinforces the hypothesis of cooperation between cells sharing the same genotype. Similarly, the emergence of cells in self-decay at the edge of two clusters is very visible in some simulations as shown in Figure 9. By doing this, those cells do not release space for cells sharing the same genotype and lose their own genotype, so this behavior is very rare and most likely usually counter selected. A hypothesis explaining the selection of such a strategy would be the creation of a barrier of cells in decay to block the progress of a hostile genotype. It is also interesting to note that during robustness testing, the genotype taking advantage early in the simulation is not necessarily the one that will dominate over the long term. This is probably explained by the progressive construction of patterns[14], as illustrated in Figure 6, and it could make HetCA an interesting model of *open-ended evolutionary developmental biology*. This reinforces the hypothesis of the existence of complex strategies HetCA, and shows that the survival phenotype changes with development. Figure 10 shows an example of the diversity of evolved strategies where the density of the blue pictured genotype is very low but it appears to be efficiently competing against the genotype in orange by massively propagating inside an orange genotype cluster when contact occurs between these two genotypes.

Further work

We have analyzed the presence of evolutionary progress in a specific category of the population: the most common genotypes. It would be interesting to analyze the potential existence of evolutionary progress in other sub categories, or indeed, in the general population. For example it is possible to hypothesize that genotype lineages using a different strategy combining low density and high robustness have been completely ignored in this analysis. Yet, in natural evolution, so-called complex living beings such as mammals, are much less numerous than prokaryotes as Escherichia coli. Seeking EP in these population groups remains to be done in HetCA. Similarly it would now be interesting to assess the influence of criteria such as the introduction of environmental change[15] on the EP.

References

Adami, C., Brown, C. T., and Haggerty, M. R. (1995). Abundance-distributions in artificial life and stochastic models:age and area revisited. In *Advances in artificial life*, pages 503–514. Springer.

Bedau, M. A. (2003). Artificial life: organization, adaptation and complexity from the bottom up. *Trends in cognitive sciences*, 7(11):505–512.

Ciliberti, S. et al. (1999). In real or artificial life, is evolutionary progress in a closed system possible? what'snew— lenski et al. *Proceedings of the Genetic and Evolutionary Computation Conference*.

Cole, B. and Muthukrishna, M. (2014). Nu-life: spontaneous dynamic hierarchical organization in a non-uniform life-like cellular automata. In *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion*, pages 99–100. ACM.

Dawkins, R. (1997). Human chauvinism.

Fitzgerald, J. and Ryan, C. (2014). On size, complexity and generalisation error in gp. In *Proceedings of the 2014 conference on Genetic and evolutionary computation*, pages 903–910. ACM.

Gould, S. J. (1988). *On replacing the idea of progress with an operational notion of directionality*. na.

Haasdijk, E., Bredeche, N., and Eiben, A. (2014). Combining environment-driven adaptation and task-driven optimisation in evolutionary robotics. *PloS one*, 9(6):e98466.

Hillis, W. D. (1990). Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena*, 42(1):228–234.

Hornby, G. S. (2006). Alps: the age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 815–822. ACM.

Johnson, N. A., Lahti, D. C., and Blumstein, D. T. (2012). Combating the assumption of evolutionary progress: lessons from the decay and loss of traits. *Evolution: Education and Outreach*, 5(1):128–138.

Lessin, D., Fussell, D., and Miikkulainen, R. (2013). Open-ended behavioral complexity for evolved virtual creatures. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 335–342. ACM.

Lynch, M. and Conery, J. S. (2003). The origins of genome complexity. *science*, 302(5649):1401–1404.

McShea, D. W. (1991). Complexity and evolution: what everybody knows. *Biology and Philosophy*, 6(3):303–324.

Medernach, D., Kowaliw, T., Ryan, C., and Doursat, R. (2013). Long-term evolutionary dynamics in heterogeneous cellular automata. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 231–238. ACM.

Ray, T. S. (1993). An evolutionary approach to synthetic biology: Zen and the art of creating life. *Artificial Life*, 1(1_2):179–209.

Shanahan, T. (2000). Evolutionary progress? *BioScience*, 50(5):451–459.

Shanahan, T. (2012). Evolutionary progress: Conceptual issues. *eLS*.

Wolfram, S. (2002). A new kind of science, champaign, il: Wolfram media. *Author Notes*.

[14] Attractors of the genotype.

[15] It would be easy to introduce environmental change such as changing the decay time of life when the cells during the simulation.

Simpson's Paradox, Co-operation and Individuality in Bacterial Biofilms

Alexandra Penn[1]

[1] Department of Sociology/Centre for Environmental Strategy, University of Surrey, Guildford UK

a.penn@surrey.ac.uk

Bacteria often live in group structures known as biofilms within which they commonly display co-operative behaviours, such as the production of public goods (Ghannoum & O'Toole 2004, Crespi 2001, West *et al.* 2007). Non-cooperative cheats arise commonly in biofilms (de Vos *et al.* 2001, Schaber *et al.* 2004), but despite what theory might predict, (Hardin 1968, Rankin *et al.* 2007), co-operation does not seem to be disrupted. The stability of these behaviours requires explanation and could cast light on the evolution of multi-cellularity experimentally (*e.g.* Rainey & Rainey 2003, Griffin *et al*. 2004, Kreft 2004, Buckling *et al.* 2007). Theory tells us that repeated aggregation into local groups, interleaved with dispersal and remixing, can increase the level of cooperation in a population despite a selective disadvantage to cooperating within any group (Wilson 1980). This increase in global proportion of co-operators despite a decrease in all local proportions, caused by the differential growth of groups, is known as Simpson's paradox (Simpson 1951). Given the microcolony (small sub-group) formation and dispersal behaviour observed in natural biofilms, it has been suggested that Simpson's paradox may explain bacterial cooperation; but although it has been demonstrated in artificially constructed groups, it has not yet been demonstrated in a natural population (Chuang et al. 2009). Using the production of siderophores in *Pseudomonas aeruginosa* as a model system for co-operation (Varma & Chincholker 2007), we measured the change frequency of co-operator and siderophore-deficient cheat strains *in-situ* within microcolony structures over time. We detected significant within-type negative density-dependent effects which vary over microcolony development. The growth of types was self-limiting at different times: Cheat growth was negatively correlated with the proportion cheats during early stages of microcolony development, with wild-type growth negatively correlated with wild-type biomass later. However, we found no evidence of Simpson's paradox (Penn et al. 2012). Instead we saw clear within-microcolony spatial structure (cheats occupying the interior portions of microcolonies) that may violate the assumption required for Simpson's paradox that group members share equally in the public good. In fact, it seems that the extent of the group over which the public good is being shared is a dynamic entity. This group, which will be defined by a lower threshold siderophore concentration for effective iron chelation, co-develops with the biofilm as the result of an interaction between population dynamics and the react-diffusion processes within it. This has interesting consequences for understanding co-operation within biofilms as well as major transitions, as the group may potentially be influenced by the bacteria themselves in order to change the context of selection and promote within-microcolony "individuality". I will discuss our observations and continuing work, both experimental and in simulation, in the broader context of a theoretical framework that suggests how factors which affect population structure, higher-level individuality and co-operative behaviour may co-evolve.

References

Buckling A, Harrison F, Vos M, Brockhurst MA, Gardner A, West SA & Griffin A (2007) Siderophore-mediated cooperation and virulence in *Psuedomonas aeruginosa. FEMS Microbial. Ecol.* **62:** 135-141

Chuang JS, Rivoire O & Leiber S (2009) Simpson's Paradox in a Synthetic Microbial System *Science* **323**: 272

Crespi BJ (2001) The evolution of social behaviour in microorganisms. *Trends Ecol. Evol.* **16:** 178-183

De Vos D, De Chial M, Cochez C, Jansen S, Tummler B, Meyer JM, Cornelis P (2001) Study of pyoverdine type and production by Psuedomonas aeruginosa isolated from cystic fibrosis patients. *Arch Microbiol* **175:**384-388

Ghannoum MA & O'Toole GA eds. (2004) Microbial biofilms. *ASM Press*. Washington, DC.

Griffin AS, West SA & Buckling A (2004) Cooperation and competition in pathogenic bacteria. *Nature* **430:** 1024-1027

Hall-Stoodley L, Costerton JW & Stoodley P (2004) Bacterial Biofilms: From the natural environment to infectious diseases. *Nature Reviews Microbiology* **2:** 95-108

Hardin, G (1968) The tragedy of the commons. *Science* **162:** 1243-148

Kreft JU (2004) Biofilms promote altruism. *Micobiology* **150:** 2751-2760

Penn AS, Conibear TCR, Watson RA, Kraaijeveld AR &.Webb JS (2012) Can Simpson's Paradox Explain Co-operation in *Pseudomonas aeruginosa* Biofilms**?** *FEMS immunology and Medical Microbiology*

Rainey PB & Rainey K (2003) Evolution of cooperation and conflict in experimental bacterial populations. *Nature* **425:** 72-74

Rankin DJ, Bargum K & Kokko H (2007) The tragedy of the commons in evolutionary biology. *Trends Ecol. Evol.* **22:** 643-651

Schaber JA, Carty NL, McDonald NA, Graham ED, Cheluvappa R, Griswold JA & Hamood A (2004) Analysis of quorum sensing-deficient clinical isolates of *Pseudomonas aeruginosa*. *Journal of Medical Microbiology* **53**: 841-853

Simpson EH (1951) The Interpretation of Interaction in Contingency Tables. *Journal of the Royal Statistical Society B* **13:** 238-241

Varma A & Chincholkar S eds. (2007) Microbial Siderophores, vol. 12 of *Soil Biology* Springer, Berlin/Heidelberg

West SA, Diggle SP, Buckling A, Gardner A & Griffen AS (2007) The social lives of microbes. *Annu. Rev. Ecol. Evol.* S. **38:** 53-77

Wilson DS (1980) The natural selection of populations and communities. *Benjamin/Cummings*, California

Emergent Robustness in Software Systems through Decentralized Adaptation: an Ecologically-Inspired ALife Approach

Franck Fleurey[1], Benoit Baudry[2], Benoit Gauzens[3], André Elie[2] and Kwaku Yeboah-Antwi[2]

[1]SINTEF, Norway
[2]INRIA, France
[3]Université de Rennes 1, France
benoit.baudry@inria.fr

Abstract

The ecosystem of web applications faces a critical paradox: on one hand, the Internet is a constantly evolving and unpredictable computing platform, on the other hand, the software services that run on top of it hardly have the ability to adapt to the evolution of this platform. Among the software services, we distinguish between *service providers* that provide micro services and *service consumers* that aggregate several micro services to deliver macro services to customers. Providers and consumers must handle uncertainty: providers cannot know in advance what consumers need; consumers rely on third-parties that can disappear at any time. Our proposal analogizes the software consumer / provider network to a bipartite ecological graph. This analogy provides the foundations for the design of EvoServ, an individual-based ALife simulator used to experiment with decentralized adaptation strategies for providers and consumers. The initial model of a software network is tuned according to observations gathered from real-world software networks. The key insights about our experiments are that, 1) we can successfully model software systems as an ALife system, and 2) we succeed in emerging a global property from local decisions: when consumers and providers adapt with local decision strategies, the global robustness of the network increases. We show that these results hold with different initial situations, different scales and different topological constraints on the network.

Introduction

The infrastructure of the Internet (computers, routers, servers and connections) can be considered an evolving complex adaptive system (Park and Willinger, 2005; Albert et al., 1999). Mapping the internet as a graph, it is possible to observe that the internet is an extremely adaptive network which is highly unpredictable due to its attrition/churn rate (i.e. nodes or connections frequently appear and disappear with no warning). Software companies tackle this unpredictability using loosely coupled architectures (Huhns and Singh, 2005): on one hand, *service providers* develop and maintain micro services that handle basic functionalities (e.g., database management, access control, etc.) which they provide over the web; on the other hand, *service consumers* access several micro services over the Internet to build macro services that they provide to customers (e.g., salary management or travel planning). Despite these efforts, both providers and consumers still face uncertainty. Providers must develop services with no certainty about what consumers exactly need, how often they need it and what level of granularity is needed. This poses an essential challenge when it comes to deciding what services to provide and in which quantities. Consumers aggregate third-party services, but they cannot predict if the provider will fail, or if the connection to the provider will fail or which provider provides the most of the services they need. In complex adaptive systems, entities that cannot evolve in response to environmental changes contribute to the imperilment of the robustness of the whole system (e.g., as demonstrated in food webs (Staniczenko et al., 2010)). In the context of software networks, this means that the lack of evolutionary capabilities in software service providers and consumers greatly reduces the robustness of the global software network (providers, consumers and connections). Today, the robustness of these software networks relies on either over-approximated redundancy (providers provide much more than needed in case the consumer's demand increases) or on centralized techniques that assume a global view and knowledge of the network and its topology. The former solution which is the most commonly used today implies and results in a waste of resources, while the latter is rarely applied because it is often impossible for software companies to build and have an accurate global view of the system (especially concerning the interaction between them and their consumers). There exists a need for novel software engineering approaches to handle the uncertainty and dynamicity of Internet applications (Bertolino et al., 2015).

In this work, we investigate a novel approach to engineering software services deployed on evolving computing platforms. Following the intuition that "*computer systems can be better understood, controlled, and developed when viewed from the perspective of living systems*" (Forrest et al., 2002), we developed EvoServ, an ecologically-inspired ALife individual-based model and simulator which models software systems as an artificial life system. EvoServ analogizes provider/consumer software interactions to mutualistic ecological interactions (e.g., bee/pollinator networks). Treating consumers and providers as individuals in an ecological system, EvoServenables the modeling of evolution in a software system during the course of its lifecycle thereby allowing the investigation of the changes in robustness of the software system during its lifecycle. We tune EvoServwith two large real-world software networks.

We propose four (4) major classes of localized strategies that describe and govern how consumers and providers evolve and adapt. These four (4) classes of strategies are generalized from the set of localized evolutionary/adaptation actions that are possible in a distributed software system. Using EvoServ, we empirically compare and contrast the adaptability of various software systems subjected to these strategies focusing on the changes in robustness of the global software systems. We define *robustness* as the ability of consumers to survive the extinction of providers and adapt the notion of extinction sequence from ecology (Burgos et al., 2007) to measure it.

Our results empirically show that, localized adaptation of consumers and providers in software systems enables the emergence of global functional robustness. They empirically prove that systems utilizing our localized adaptation strategies perform better than a random adaptation strategy (33% robustness with random adaptation vs. 46% with our adaptation strategies). We also run experiments that confirm the stability of these results at different scales and on different software systems.

In this paper, we develop three main contributions

- EvoServ[1], an ALife, individual-based simulator that accurately models the evolution and adaptation of a complex software system during its lifecycle and allows the investigation of the resultant robustness.

- Empirical evidence showing that global robustness can emerge from localized evolution rules in provider/consumer software networks.

- Empirical data about two real-world provider/consumer software graphs in which we consistently observe a power-law distribution for (i) the number of services in consumers and (ii) the rate of service usage among consumers. We use this data to tune the initial bipartite network used in EvoServ.

Bipartite interactions in Internet Computing

In this section, we introduce the bipartite graph model upon which our simulation is built, as well as the different evolutionary strategies and mechanisms that we experiment with. We chose a bipartite graph model as the foundation because it allows us to capture and represent the interactions and relationships present in a software network.

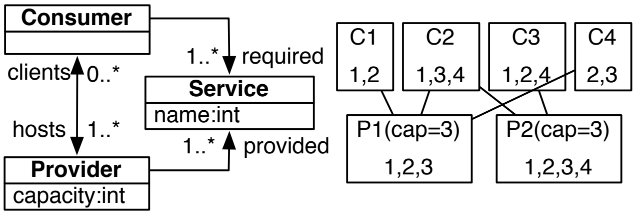


Figure 1: Model and one instance of software bipartite graph

Bipartite software interactions

The left part of Fig. 1 describes the data model we use to abstract bipartite software relationships between the different components of a software system.

- `Provider` individuals are an abstract representation of nodes on the Internet that provide micro-services to other nodes (e.g. a web server). A `Provider` P, is characterized by three attributes: `provided`, a list of services provided by P; `clients`, the list of consumers that consume at least one service of P; a `capacity`, the maximum number of Consumers that can access one or more of P's services (every individual accessing P is counted once irrespective of the number of services accessed).

- `Consumer` individuals aggregate remote services in order to offer macro-services to their customers (e.g., a travel web site such as Expedia). A `Consumer` C is characterized by two attributes: `required`, a list that contains all the services that C needs; `hosts`, the set of providers to whom C connects to over the Internet in order to get all the services it needs (we do not impose a limit on the number of providers to whom a consumer can connect to).

A `Service` in our model refers to a resource that is being required or provided. A service is referenced by its name(an integer identifier in our model).

The right part of Fig. 1 displays an example of a software system modeled in EvoServ. It contains 4 consumers and 2 providers with both providers having a capacity of 3. Consumer $C1$ requires services 1 and 2 to function correctly and accesses those services through a connection to $P1$; $C2$ requires services 1, 3 and 4 and accesses them through connections to $P1$ and $P2$. Provider $P1$ is currently at maximum capacity while $P2$ can still serve one more client.

The following sub-sections describe the dynamic aspects of software bi-partite graphs. We distinguish between two types of changes: adaptation and evolution. *Adaptation* refers to changes to individuals nodes and it does not vary the composition of the set of nodes in the system, while *evolution* refers to changes which vary the composition of the system(ie. providers and consumers can appear and disappear over time).

[1] `https://github.com/DIVERSIFY-project/EVOSERV`

Adaptation in EvoServ

Provider Adaptation Providers adapt by changing the set of services they offer. We experiment with two different types of adaptation operators

Random: A provider P chooses at random to either add a service or drop one of the services it provides.

Popular: A provider P chooses either to add a popular service to its list of services provided or to drop a unpopular service from its list of services provided. When P adds a service, it favors services that are popular among its consumers with more popular services having higher probabilities of being selected. Conversely, when P drops a service, services that are not popular among P's consumers have higher probabilities of being dropped. The probability of dropping a service increases with the number of consumers connected to P. This is in order to reduce the pressure on this provider when it's at capacity. We assume that P can access the set of services required by its consumers.

Constraints on Provider's adaptive behavior: Both adaptive behaviors described above are constrained such that, all consumers of a given provider, P, can always access what they require. Thus, if P decides to drop a Service, it first checks that none of its consumers would go extinct (each consumer provides a routine that assesses whether the Service is strictly required).

Consumer Adaptation Consumers adapt by changing the links to providers through which they access their required services. We experiment with two different kinds of adaptive operators

Random: A consumer, C, chooses at random a new provider to link to.

Equitable: A consumer, C, attempts to optimize the diversity of its service provisioning by accessing a set of providers that maximizes its equitability (`i.e.` the number of times each required service is provided). Given $\mathcal{S} = \{s_1, \ldots, s_n\}$, the set of services that a consumer requires and $\#occ_{s_i}$, the number of times a service s_i is provided through a link to a consumer, the Shannon index for a consumer is:

$$H'_{i,\mathcal{P}} = -\sum_{k=1}^{n} \frac{\#occ_{s_k}}{n} \ln \frac{\#occ_{n_k}}{n}$$

and its equitability is $\exp(H'_{i,\mathcal{P}})$. Maximizing equitability ensures that a consumer has a set of connections which offers each required service an even number of times.The concept of equitability is taken from ecology where it is used to evaluate biodiversity.

In practice, many different advertisement or discovery mechanisms can be used to find the set of potential providers. Our experiment does not intend to replicate any particular mechanism but only assumes that at any point in time, consumers have access to a neighbourhood of providers, which we pick as a random subset of the set of all providers.

Constraints on consumers adaptation: Both adaptation operators described above are constrained such that, each required service that was already provided remains provided. Any existing link that only provides consumer C with services that are also provided through other links can be discarded, thereby allowing new links to be made to any provider in the neighbourhood who provides at least one of C's required services thats currently unprovided.

Global constraints over the model to keep costs comparable between different experiments :

The total number of links in the network or the total number of services provided in the network can change over the adaptation process and can introduce a confounding bias. Allowing more links or introducing more provided services can have a positive impact on robustness by providing consumers with more opportunities to link to their required services. However, we wanted to prevent this phenomenon in order to isolate the effect of adaptation rules when evaluating the impact on global robustness.

The number of services and links are therefore kept constant. This constraint is enforced in the following manner: the simulator allows a random set of nodes to run adaptations which decrease the number of links and services and collects tokens corresponding to these deletions. In a second step, those tokens are distributed randomly in order to allow some nodes to add links and services. This token system ensures the decentralized nature of our simulations while conserving some global constraints. In the real world, this corresponds to fixing the total amount of available resources in the software system (e.g., the number of services and the amount of bandwidth).

Ecological concepts for robust software networks

In this work, we analogize a network of software services that run over the Internet to bipartite ecological graphs. This analogy comes with two essential principles that we build upon in this work: 1) species in ecological communities adapt in a completely decentralized manner to cope with a continuously evolving environment; 2) ecologists have developed sound measures to quantify the robustness of a bipartite graph as represented by interactions between species. In the following, we summarize these two ecological principles.

Species adaptation in ecosystems is completely decentralized and is not driven by a global goal at the ecosystem level. Adaptation is necessary in the face of an evolving environment; as environmental conditions change over time, species have to adapt (i.e. keep a good reproductive success) to their local conditions. The *Red Queen hypothesis* (Van Valen, 1973), named in reference to a statement[2] made to Alice by the Red Queen in Lewis Carroll's *Through the looking glass*, crystallizes the idea that, species need to constantly evolve in order to survive, in the same way as Alice and the Red Queen need to constantly run to stay in the same place. The *Red Queen Hypothesis* provides a good theoretical `raison d'être` for adaptation, stating that, in order to increase the persistence of our Consumers and Providers individually, a constant adaptation is key to keeping a good fit to specific conditions. This hypothesis however lacks a comprehensive view at the macroscopic scale: is the robustness of our system an emergent property of such behavior?

Some experiments have shown that large levels of diversity and redundancy promote the stability of functional processes in ecosystems like biomass production (Tilman and Downing, 1994). However, the mechanisms underlying this experimental evidence remains elusive in the case of complex networks of interacting species. Mathematical models show that more complex systems (in terms of species richness and number of interactions) are more likely to collapse. This incompatibility between theoretical and empirical observations is known as May's paradox (May, 1972). This paradox and more broadly, the relationship between complexity, evolution and stability is a still open fundamental question in ecology. Most of this previous work reasons on ecological graphs where nodes represent species, and edges ecological interactions (e.g. predation or parasitism). The topology of ecological graphs (emerging from ecological processes such as extinction, colonization, but also evolution) is non random, and its structure seems to favour stability (Yodzis, 1981).

We investigate evolution and adaptation in software systems in order to experiment with the emergence of global robustness through localized decentralized actions. Given a bipartite graph that models a network in which nodes are species and edges are species interactions, the *robustness measure* (Burgos et al., 2007) quantifies the ability of one level of the graph to survive the extinction of species in the other level. The primary extinction sequence can be performed according to different strategies: randomly removing nodes, removing the most or least connected ones, etc. In the case of mutualistic graphs, as soon as a species looses all of its connections, it is unable to reproduce (for plants) or feed (for pollinators) and thus goes extinct. This is called a *secondary extinction*. Referring back to the network in Fig. 1: if P1 is removed, C4 goes extinct. If C1 has the ability to adapt, it can connect to P2 otherwise it also goes extinct, a *secondary extinction*; subsequently, when P2 is removed, the whole network goes extinct.

Fig. 2 plots the relationship between secondary and primary extinctions. The robustness index is the area under the curve. In our experiments, we normalize the robustness index by considering it a ratio of the maximum robustness value (in Fig. 2 the robustness is thus $\frac{area\_under\_curve}{350\times100}$). Since primary extinctions are random, we compute several robustness indices on a given graph to obtain a mean. For example, Fig. 2 shows 20 runs and the mean value as a thick line: the lowest robustness index is 0.33 and the max is 0.4.

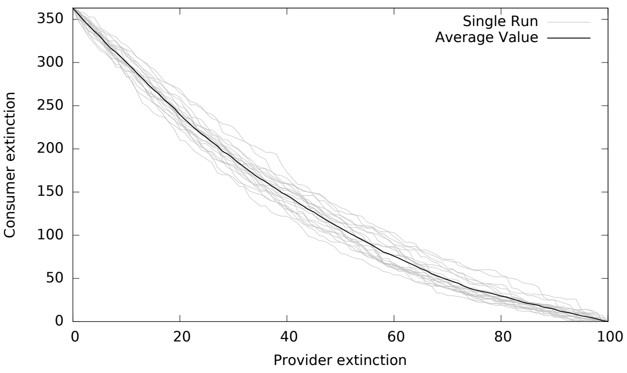


Figure 2: Relationship between secondary and primary extinctions in a bipartite graph (with a random primary extinction sequence)

Experimental design

All of our experiments start with a bipartite graph that models a software system and then simulates the evolution of that graph to evaluate the effect of the decentralized adaptation rules. In this section, we discuss how we set up the initial graph, and how we tune the different experimental parameters.

Mining software interactions to tune EvoServ

We analyzed two real-world software systems in order to generate real world data to aid us in tuning of our initial graph. We collected data about installations of WordPress [3], an open-source content management system used to easily deploy web sites, and also data about web browsers. We selected these two case studies because both are very popular technologies (WordPress is deployed on 23% of the top 10 million web sites [4], and billions of web browsers are deployed and used daily worldwide) and also because both technologies are open and can be easily extended by their users through the addition of plugins that provide specific functionalities (e.g., display photo, read pdf, etc.). For both case studies, we collected the number of plugins installed on each client in the dataset and the subsequent distribution of plugin usage. This data is used to tune the size of software entities in our graph, as well as the distribution of services on these entities.

[2] "Now, here, you see, it takes all the running you can do, to keep in the same place."

[3] `https://wordpress.org/`

[4] `http://w3techs.com/technologies/overview/content_management/all/`

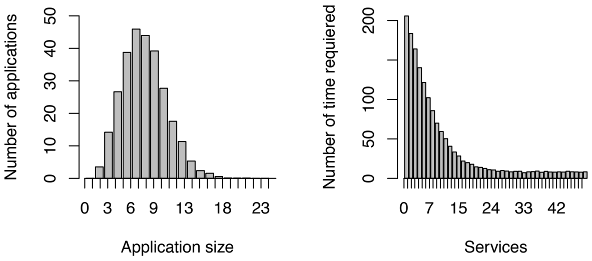


Figure 3: Number of Services per entity (left) and distribution of Service usage (right) in the initial graph

WordPress: Analyzing the 500 000 top web sites [5], we selected the ones that use WordPress. This gave a list of 110 000 WordPress sites. We then crawled these sites to find out which plugins they use. Analyzing this data [6], we found out that the number of plugins per site follows a poisson distribution with an average value of 5. The distribution of plugin usage follows a negative exponential slope.

Web browsers: Since browsers are installed on user machines, we could not access this data directly. We therefore set up `https://amiunique.org/`, a website where we collect anonymous information about the browser, the list of fonts and plugins it contains from every visitor. By February 2015, the site had been visited by 63,000 unique visitors. All of the observed browsers contained a total of 1,920 different plugins. We observeed distributions very similar to the ones in WordPress: a negative exponential slope for the plugin usage distribution and poisson distribution with an average of 6 for the number of plugins per browser.

Tuning the initial model The initial graph for our simulation had the following size: 300 Consumers, 100 Producers and 50 Services. The 1:3 ratio between the 2 first parameters was loosely based on a Cloud Service Brokerage example, with the values increased by an arbitrary multiplying factor of 100. We fixed the number of evolution cycles through which the bipartite graph is run to 500. We then used the data observed on real systems to tune the size of Consumers, as well as the distribution of the Service usage among these Consumers. As seen in Fig. 3, the size of Consumers follows a poisson distribution of parameter 5.47, and the Service usage, a power law distribution of parameter 2.08.

Experiments

In order to evaluate the effect of the adaptation strategies described in section II, we run the simulation with four different combinations of adaptive behaviors for Consumers and Providers.

random-random: Consumers and Providers adapt randomly. This is the baseline

random-popular: Consumers adapt randomly and Providers adapt according to the *popular* adaptation behavior.

equitable-random: Consumers adapt according to the *equitable* adaptive behavior and Providers adapt randomly.

equitable-popular: Consumers adapt according to the *equitable* adaptive behavior and Providers adapt according to the *popular* adaptation behavior.

To ensure statistical validity, we needed to run the simulation and robustness computations a sufficient number of times. We utilized Monte Carlo estimation to determine the number of times needed.

- We determined that the robustness value of one graph should be the mean value computed over 50 extinction sequences (each sequence randomly picks providers that go extinct). Monte Carlo methods showed that, after 50 sequences, the variance of a new robustness value was below 0.05%

- We run each simulation over 50 different initial graphs and determined that, after 50 graphs, the variance for results from a new simulation was below 0.01% .

Results

RQ1. Can local adaptation lead to the emergence of global robustness in software systems?

This was the key research question for our work. We evaluated the impact of the different adaptation strategies on the evolution of global system robustness. The "random-random" strategy was used as a baseline, mimicking how software systems currently do adapt. All of the adaptation strategies were subject to the same global constraints in the model; `i.e.` all consumers remain satisfied and the cost of the model (total number of services provided and total number of links) is kept constant.

Fig. 4 shows the evolution of the robustness of the global system for the four strategies. This plot was obtained by averaging the results from 50 runs of the simulator on 50 randomly generated initial models. Table 1 presents the values for the average final robustness for the different strategies and their standard deviation.

These results show that, the localized adaptation operators had a positive impact on the global robustness of the system when only one class of nodes(Providers and Consumers) was utilizing them (rows 2 and 3 in Table 1). More interestingly, the hybrid approach of all nodes utilizing their localized adaptation operators simultaneously(row 4 in Table 1) resulted in a much higher benefit than when only one was. This shows that there is a synergy between those decentralized strategies and this is in favor of the global robustness

[5] `http://www.alexa.com/topsites`

[6] Wordpress data available here: `http://diversify-project.eu/wordpress/`

Adaptation Strategy		Robustness	
Consumer	Providers	Average	Std. Dev.
Random	Random	33.4%	0.896
Equitable	Random	37.7%	1.084
Random	Popular	35.9%	1.018
Equitable	Popular	46.2%	1.584

Table 1: Final robustness of the global system with four different adaptation strategies (observations on 25 runs)

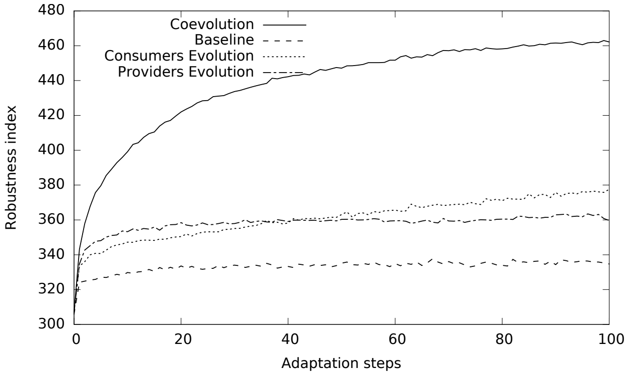


Figure 4: Evolution of globalized robustness using different decentralized adaptation strategies

of the system. This result is interesting because the strategies used were only based on local knowledge and target local optimizations for individual Providers and Consumers. There was no explicit push for these individual strategies to have a positive effect on the global robustness of the system.

Although the results from the simulations described so far are encouraging, we can only conclude from them that global robustness can emerge in theese specific simulation scenarios. In practice, a number of simplified decisions were made in the simulations. Firstly, the Providers and Consumers were constant. A real world system is more dynamic with Providers and Consumers constantly appearing and disappearing and becoming either temporarily or permanently unavailable. Secondly, the distribution of Providers and Consumers and the distribution of Services were constant but may vary in real life. We decided to investigate the results of the simulation in the absence of such simplifications and also investigated if the results would hold for systems of a different scale. The next 3 research questions discuss these investigations.

RQ2. How sensitive are the adaptation rules when relaxing topological constraints of the software system?

This question focuses on the impact of the hard constraints of previous experiments (fixed size and topology of graph) on our results. We ran a set of experiments where Consumers and Providers evolved randomly (they could go extinct, be mutated, reproduce or be cloned allowing some new entities to appear) in parallel to the adaptation strategy. Reproduction, cloning and extinction probabilities were set at 30%. New Consumers were produced by crossover of two ancestors with the new node containing a sub-set of the services of its ancestors required. New Providers were created using one of the two evolution strategies described below.

Providers' random evolution strategies : New Providers are created by cloning an existing one and mutating the list of provided Services (add a new Service or remove one). Service mutation probability was set at 20%.

Baseline Random adaptation with random evolution of Providers.

Equitable popular Equitable-Popular adaptation with a random evolution of Providers.

Providers' ecological evolution strategies : This improves the random evolution strategy such that, when cloning a Provider, the mutation factor is no longer a static probability. The mutation rate of a provided Service depends on its success among the Consumers connected to it: the more successful a service, the greater the probability it will also be provided by the clone. This strategy introduces a form of environmental pressure in the adaptation, with the aim of creating offsprings that are more fit than their ancestors.

Baseline (ecology). Random adaptation with an ecological evolution of Providers.

Equitable popular (ecology). Equitable-Popular adaptation with an ecological evolution of Providers.

Fig. 5 shows the results for 4 experiments which combine the two adaptation strategies (Random and Equitable-Popular) with the two alternative evolutionary strategies: a fully random one and an ecology inspired one, which is shown to bring improvement slower than the original, but eventually to surpass it.

The impact on robustness was similar to the previous, more constrained, experiments. The main difference is that the rate of convergence is slower: the results of Fig. 5 are for 500 combined adaptation and evolution steps while on Fig. 4 only 100 steps were necessary when no evolution was simulated.

RQ3. How does the distribution of services among consumers influence the results?

The distribution of services among the Consumers is a critical parameter of the simulation. In all previous experiments, the size of Consumers followed a poisson distribution of parameter 5.47 and the Service usage, a power law distribution of parameter 2.08. Since these distributions were tuned by observing real bipartite software interactions, we are confident about their relevance. Yet, the exact parameters of

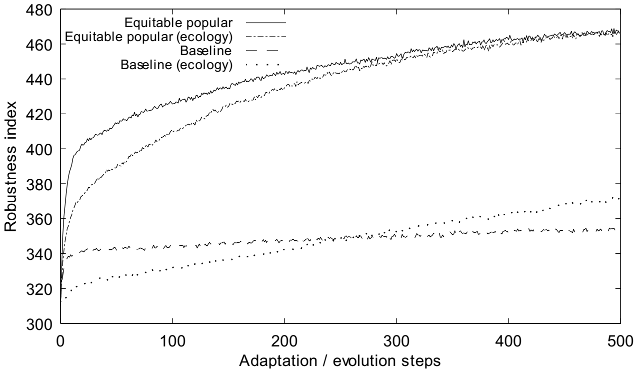


Figure 5: Evolution of the overall robustness using different decentralized adaptation strategies in parallel with different system evolution.

those distributions will vary from one situation to the next (and even possibly over time).

This question addresses the sensitivity of the results with respect to these distributions. In order to do answer it, we replicated the simulation presented on Fig. 4 using graphs generated from different statistical distributions.

Table 2: Results for different initial distributions of nodes sizes and Services usage. P(λ) for a Poisson distribution, N(μ, σ) for a normal distribution, E(λ, min) for a power law with a minimum value of min and NE(λ, u) for a Negative Exponential distribution with an added constant u.

D_Size	D_Services	Init.	Random	Eq-Pop	Delta
P(3)	NE(0.25,0.005)	35.9	44.8	47	**2.2**
P(6)	NE(0.25,0.005)	39.8	33.3	46.9	**13.6**
P(6)	NE(0.25,0.001)	35.5	39.1	51.9	**12.8**
P(6)	NE(0.25,0.01)	28	31.2	42.9	**11.7**
P(6)	UNIFORM	21.6	27.7	32.4	**4.7**
P(9)	NE(0.25,0.005)	27.7	27.8	44.5	**16.7**
E(8,1)	NE(0.25,0.005)	37.6	38.9	47.8	**8.9**
E(15,3)	NE(0.25,0.005)	32.6	34.4	47.5	**13.1**
N(5,5)	NE(0.25,0.005)	33.4	35.3	47.7	**10.4**

Table 2 presents the results of the simulations. The first and second columns liindicates the parameters for the distribution of Consumer size (D_{size}) and Service usage ($D_{services}$) respectively. The next columns list the robustness value of the initial network ($Init.$) and of the final network evolved with the baseline random ($Rand.$) and the equitable-popular ($Eq - Pop.$) strategies. The last column indicates the difference in final robustness between both adaptation strategies. The results show that the guided strategy always performs significantly better than the baseline. The two lowest results (row 1 and 5) correspond to extreme situations. In row 1, a P(3) distribution makes the average size of Consumers 3, with a good portion of Providers and Consumers of 1 or 2. In that case, the exploration space is very much reduced and the impact of the guided strategy is limited. In row 5 the Services are chosen using a uniform distribution: the nodes are provided with sets of Services that are completely independent from one another. When using a power law (all other rows in the table), the sets of Services are related and more similar. In practice, the uniform distribution is unrealistic: there always are very popular micro-services used by everyone (e.g., Google Analytics).

RQ4. What is the effect of the system's scale?

To assess the generality of the results beyond the fixed network size used in previous experiments (300 Consumers, 100 Producers and 50 Services), we repeated them at different scales and using different ratios between the values. All experiments have shown a clear benefit of the Equitability-Popular strategy over the baseline with various degrees. For example, repeating the experiments with a number of 5000 Consumers and 1000 Providers yielded an initial robustness of 40.3 and a final average robustness of respectively 43.3 and 52.2 for the baseline and Equitability-Popular. With respect to scalability and sensibility to the different parameters, the Equitable-Popular strategy was always significantly better than the random strategy.

Related works

Lim and Bentley (2012) propose an ALife agent-based model to simulate the evolution of an ecosystem of software apps on an app store. They model app developers, app users and apps as agents and experiment with different behaviors for developers. Their goal is to determine what are the best strategies for app developers in order to create successful apps. They manage to identify a combination of strategies and frequency for each strategy with which they can reproduce the development Apple's iOS app ecosystem. Cocco et al. (2014) have recently provided similar modeling to understand what strategies should be adopted by application developers. While these works adopt a model similar to our, their goal is different: understand good strategies for developers in their case, model good adaptation strategies for global robustness in our case.

Goings et al. (2012) present an ecology-based evolution algorithm that produces good behavioral models for complex software systems. They show that adapting ecological principles for evolution produces models of software that are more diverse and evolvable than the models produced by a more classical evolutionary algorithm. Similarly to our simulation, their evolutionary algorithm evolves models in a very constrained environment (fixed population and limited resources), which eventually pushes towards more diversity and good properties for future evolutions.

Another area of related work is the simulation of service-based systems (Wang et al., 2010; Kaur et al., 2013). This is a very active area as shown in the literature reviewed by

Smit and Stroulia (2013). They review 6 major frameworks for the simulation of service-oriented systems. Simulation are either based on the abstract description of the services or on more formal descriptions that rely on discrete-event semantics. These different models support different tasks such as testing, interaction vizualization, code generation or performance prediction. However, none of these frameworks evaluates the impact of perturbations, nor simulates decentralized evolution. Other works have evaluated the ability of service-oriented systems to sustain perturbations, such as denial-of-service attacks (Xu and Lee, 2003), but did not reason about the global dynamics of the system to establish robustness.

Conclusion

In this paper we have proposed EvoServ, a simulation model to experiment with decentralized adaptation strategies in distributed software systems. This simulation model is calibrated with data from actual software models. We have proposed a set of guided decentralized adaptation strategies and evaluated them against a random baseline in a set of different simulations. The results indicate that the combinations of the proposed Equitable and Popular local strategies significantly increases ability of the system to sustain the extinction of service Providers (i.e., called robustness in this paper). These results mark a significant first step towards the transposition of the ecological *Red Queen hypothesis* into actual software systems. Future work include two main threads: analyze topological properties of the graph over the simulation to characterize the phenomena that favor robustness emergence; deploy these behaviors on a controlled SOA system.

In the future, developers of platforms that provide microservices and of service mash-ups that aggregate services could enhance their products with simple decision making strategies, which benefit the local nodes and participate in a system-wide improvement of robustness.

Acknowledgements

This work is partially supported by the EU FP7-ICT-2011-9 No. 600654 DIVERSIFY project.

References

Albert, R., Jeong, H., and Barabási, A.-L. (1999). Internet: Diameter of the world-wide web. *Nature*, 401(6749):130–131.

Bertolino, A., Blake, B., Mehra, P., Mei, H., and Xie, T. (2015). software engineering for internet computing. *IEEE Software*, 32.

Brännström, A., Johansson, J., Loeuille, N., Kristensen, N., Troost, T. A., Lambers, R., and Dieckmann, U. (2012). Modelling the ecology and evolution of communities: a review of past achievements, current efforts, and future promises. *Evolutionary Ecology . . .*, 14:601–625.

Burgos, E., Ceva, H., Perazzo, R. P. J., Devoto, M., Medan, D., Zimmermann, M., and María Delbue, A. (2007). Why nestedness in mutualistic networks? *Journal of theoretical biology*, 249(2):307–13.

Cocco, L., Mannaro, K., Concas, G., and Marchesi, M. (2014). Simulation of the best ranking algorithms for an app store. In *Mobile Web Information Systems*, pages 233–247.

Forrest, S., Balthrop, J., Glickman, M., and Ackley, D. (2002). Computation in the wild. In Park and Willinger (2005).

Goings, S., Goldsby, H., Cheng, B. H. C., and Ofria, C. (2012). An ecology-based evolutionary algorithm to evolve solutions to complex problems. In *Proc. of the Int. Conf. on the Simulation and Synthesis of Living Systems (ALife)*.

Huhns, M. N. and Singh, M. P. (2005). Service-oriented computing: Key concepts and principles. *Internet Computing, IEEE*, 9(1):75–81.

Kaur, N., McLeod, C., Jain, A., Harrison, R., Ahmad, B., Colombo, A. W., and Delsing, J. (2013). Design and simulation of a soa-based system of systems for automation in the residential sector. In *Proc. of the International Conference onIndustrial Technology (ICIT)*, pages 1976–1981.

Lim, S. L. and Bentley, P. J. (2012). How to be a successful app developer: Lessons from the simulation of an app ecosystem. *SIGEVOlution*, 6(1):2–15.

May, R. M. (1972). Will a Large Complex System be Stable? *Nature*, 238(5364):413–414.

Park, K. and Willinger, W. (2005). *The Internet as a large-scale complex system*, volume 3. Oxford University Press.

Rossberg, a. G., Matsuda, H., Amemiya, T., and Itoh, K. (2006). Food webs: experts consuming families of experts. *Journal of theoretical biology*, 241(3):552–63.

Smit, M. and Stroulia, E. (2013). Simulating service-oriented systems: A survey and the services-aware simulation framework. *Services Computing, IEEE Transactions on*, 6(4):443–456.

Staniczenko, P. P. A., Lewis, O. T., Jones, N. S., and Reed-Tsochas, F. (2010). Structural dynamics and robustness of food webs. *Ecology Letters*, 13(7):891–899.

Tilman, D. and Downing, J. A. (1994). Biodiversity and stability in grasslands. *Nature*, 367(6461):363–365.

Van Valen, L. (1973). A new evolutionary law. *Evolutionary theory*, 30:1–30.

Wang, W., Wang, W., Zhu, Y., and Li, Q. (2010). Service-oriented simulation framework: An overview and unifying methodology. *Simulation*, page 0037549710391838.

Xu, J. and Lee, W. (2003). Sustaining availability of web services under distributed denial of service attacks. *IEEE Transactions on Computers*, 52(2):195–208.

Yachi, S. and Loreau, M. (1999). Biodiversity and ecosystem productivity in a fluctuating environment: the insurance hypothesis. *Proceedings of the National Academy of Sciences of the United States of America*, 96(4):1463–8.

Yodzis, P. (1981). The stability of real ecosystems. *Nature*, 289:674–676.

Steering a Complex Adaptive System: A Complexity Science Design Methodology Applied to an Industrial Ecosystem in the Humber Region, UK

Alexandra S. Penn[1]

[1] Evolution and Resilience of Industrial Ecosystems (ERIE) Project, Centre for Environmental Strategy/Dept. of Sociology,
University of Surrey
a.penn@surrey.ac.uk

Many important challenges facing society today involve the management of interlinked complex adaptive systems (CAS): coupled socio-economic and ecological systems composed of many interacting elements which have been created or partially created by human actions. As we explicitly wish to manage and transform these systems, engineering and design approaches have much to offer us, however they must be fundamentally modified to deal with CAS. These systems are not static artifacts, but dynamic, evolving and reflexive *processes* the behaviour of which is not straightforwardly predicable and which may respond in unexpected ways in response to our interventions. Additionally many of the complex systems which we would most like to influence have significant social components. Objective choices about design goals cannot be made and the integration of participatory or political processes may be required.

In order to manage complex adaptive systems, we suggest a "steering" approach; an action or series of actions applied to a complex system and/or its environment for achieving a specific purpose. Steering combines tools from complexity science with whole systems design philosophy and is a continuous process which involves interacting with, monitoring and learning from the system in question. The techniques required for effective steering fall into two categories. Firstly we wish to understand, and indeed exploit, the systems' structure and dynamics in order to intervene efficaciously with them. Hence we need techniques to uncover this structure and to choose points of intervention: system "levers" through which the system as a whole could be manipulated with system interventions designed accordingly. Secondly we frame those techniques within a participatory "adaptive management" structure (Waltner-Toews and Kay, 2005), which explicitly takes into account the adaptive nature of these systems and our limited capacity to fully model real world complex systems, by building in monitoring and feedback processes with which to modify our interventions as systems respond.

We are currently applying this process to a case study aiming to facilitate regional decision making in an industrial ecosystem in the Humber region, UK. The region represents a significant proportion of UK infrastructure, energy generation capacity and CO_2 production. However, there is a strong desire to develop the "Humber Gateway" as a renewable energy hub using the extensive agricultural hinterland and offshore and port facilities to support bio-based energy production and offshore wind and tidal generation. We have undertaken participatory modelling exercises in which stakeholders collaboratively constructed simple systems models of the development of their regional bio-based economy, the key factors of influence, drivers and their perceived interdependencies (Penn *et al*., 2013). Building on this approach we used a "control nodes" methodology from network theory (Liu *et al*., 2011) to determine the specific subsets of these factors which could theoretically be used to drive the system into any given state. This technique is combined with an evaluation of the practical controllability of each factor from the perspectives of the actors present to allow its use in real world contexts in which policy makers and industrial stakeholders must make decisions (Penn *et al*., In press.).

Applying this hybrid approach in the real world context of the Humber allowed stakeholders to uncover and discuss possible novel points of intervention in their regional system, based both on its structure and their own differing abilities to influence different factors within it. This proved to be a useful starting point for debate on policy ideas and the importance of considering not just *where* to intervene, but *who* is able to intervene with a given factor. However this approach, like all network-based methods, is limited by its sensitivity to the network structure described. Additionally by the fact that once suitable control configurations are discovered, the method as yet gives us no indication of how to construct the time variable control inputs required to steer the system to a given state. Instead we must use such modelling as a "thinking tool" to provide principled starting points, to be combined with stakeholder expertise, for the participatory design of complex social systems.

References

Liu, Y.Y., Slotine, J.J., and Barabasi, A.L. (2011) Controllability of complex networks, *Nature* 473, pp. 167-173.

Penn A.S., Knight, C.J.K., Lloyd, D.J.B., Avitabile, D., Kok, K., Schiller, F., Woodward, A., Druckman, A., and Basson, L. **(**2013**)** Participatory development and analysis of a fuzzy cognitive map of the establishment of a bio-based economy in the Humber region. *PLoS one* 8(11) DOI: 10.1371/journal.pone.0078319

Penn A.S, Knight, C.J.K., Chalkias, Y.,Velenturf, A., Lloyd, D.J.B. (*In press*) Extending a Fuzzy Cognitive Map Using Control Nodes: a case study of influencing the development of a bio-based economy in the Humber region. In *Participatory Modeling in Environmental Decision-making: Methods, tools, and applications*. Springer. *Eds* Gray, S., Gray, S., and Jordan, R.

Waltner-Toews, D., and Kay, J. (2005). The evolution of an ecosystem approach: the diamond schematic and an adaptive methodology for ecosystem sustainability and health. *Ecology and Society* 10(1): 38.

Recombination Is Surprisingly Constructive for Artificial Gene Regulatory Networks in the Context of Selection for Developmental Stability

Yifei Wang[1], Yinghong Lan[2], Daniel M. Weinreich[2], Nicholas K. Priest[3] and Joanna J. Bryson[1]

[1]Intelligent Systems Group, Department of Computer Science, University of Bath, Bath, BA2 7AY, UK
[2]Department of Ecology and Evolutionary Biology, Brown University, Providence, RI 02912, USA
[3]Department of Biology and Biochemistry, University of Bath, Bath, BA2 7AY, UK
yifei.wang@computer.org; yinghong_lan, daniel_weinreich@brown.edu; n.priest, j.j.bryson@bath.ac.uk

Abstract

Recombination is ubiquitous in multicellular plants, animals and even fungi. Many studies have shown that recombination can generate a great amount of genetic innovations, but it is also believed to damage well-adapted lineages, causing debates over how organisms cope with such disruptions. Using an established model of artificial gene regulatory networks, here we show that recombination may not be as destructive as expected. Provided only that there is selection for developmental stability, recombination can establish and maintain lineages with reliably better phenotypes compared to asexual reproduction. Contrary to expectation, this does not appear to be a simple side effect of higher levels of variation. A simple model of the underlying dynamics demonstrates a surprisingly high robustness in these lineages against the disruption caused by recombination. Contrary to expectation, lineages subject to recombination are less likely to produce offspring suffering truncation selection for instability than asexual lineages subject to simple mutation. These findings indicate the fundamental differences between recombination and high mutation rates, which has important implications for understanding both biological innovation and hierarchically structured models of machine learning.

Introduction

Understanding the introduction and maintenance of innovation in evolutionary systems is of critical interest in both artificial intelligence and evolutionary biology (Hu et al., 2014; Payne et al., 2014). Yet even basic questions such as explaining the costs and benefits of sexual reproduction are still unknown to both biological and computational science. Sex implies recombination — the reshuffling of parental genetic information, which generates heritable innovations (Eshel and Feldman, 1970; Feldman et al., 1996; Otto and Feldman, 1997; West et al., 1999). However, sexual reproduction is also considered to be very costly since it may damage well-adapted lineages, and necessarily produces fewer directly reproductive offspring, since it also produces males. Evolution should favour defection to a lower-cost strategy, such as asexual reproduction. How then is sexual reproduction maintained?

For decades, researchers have been making tremendous efforts and proposing numerous theories to uncover the mystery of sex and recombination (Eshel and Feldman, 1970; Hurst and Peck, 1996; West et al., 1999; Otto and Lenormand, 2002; Meirmans and Strand, 2010; Wagner, 2011). Two classic benefits of sexual reproduction are nevertheless still controversial: 1) purging deleterious mutations more efficiently, and 2) creating novel gene combinations (Kondrashov, 1993; Otto and Feldman, 1997; Otto and Gerstein, 2006; Kouyos et al., 2007; Barton, 2009; Martin and Wagner, 2009). An important third possibility is that the process of recombination, by allowing the localisation of both coherence and variation across the genomes of a population, is able to both improve robustness and facilitate evolutionary adaptation, a process known as *evolvability* (Wang et al., 2014). Although robustness and facilitated adaptation are observed phenomena, and are often attributed to sexual reproduction, the underlying mechanisms are still poorly understood (Wagner, 2011).

Recently, a gene regulatory network (GRN) model first proposed by Wagner (1994, 1996) has emerged as a popular computational approach to study recombination in a network context (Azevedo et al., 2006; MacCarthy and Bergman, 2007; Lohaus et al., 2010; Le Cunff and Pakdaman, 2014). One novel feature in Wagner's GRN model is that it introduces selection for phenotypic stability, performed as a separate layer of truncation selection in addition to selection for a particular optimal phenotype. Because of this truncation selection imposed on the population, only individuals that can achieve developmental equilibrium (reaching a stable phenotypic pattern) are able to survive during evolution. The central assumption is that an individual's phenotype should be able to buffer against genotypic variations (Wagner, 1996). In other words, stability selection provides a viable simulation for the known natural phenomenon of an individual's phenotype being expressed relatively stably while its genotype undergoes evolution.

An interesting feature of studying evolution in artificial gene regulatory networks is that we can find clear evidence that evolution is not a simple optimisation process. The shifting genetic characteristics of the population resulting from mutation and selection optimise and innovate, but in

Nature the optima they track are also transient. Consequently, goals for an evolutionary genome always include robustness and agility, not only gradient ascent. In a previous study, Siegal and Bergman (2002) showed that even without significant selection towards an optimal phenotype, lineages are still able to move towards the optimum so long as the population is under selection for developmental stability. Although this suggests stability selection is an important evolutionary force, the role of recombination is left unclear since the earlier simulations ignored the possibility of asexual populations. In a different study, Azevedo et al. (2006) and Lohaus et al. (2010) discovered that sexually reproducing organisms evolved higher mutational and recombinational robustness than asexual lineages. But these authors did not measure the phenotypic distance of evolved asexual and sexual populations from the optimum. Therefore, it is an open question whether recombination is still able to sustain sexual reproduction as lineages near a fitness peak.

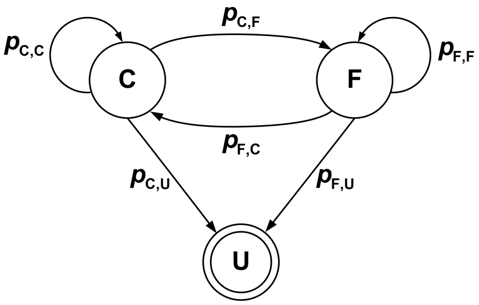


Figure 1: State transitions in an artificial gene regulatory network. There are three states in the system: **C**: individuals that are close to the optimum, **F**: individuals that are far from the optimum, and **U**: individuals that are unable to achieve developmental stability. **U** is an absorbing state in the system, since unstable genomes cannot reproduce.

To get a better intuition for the questions of evolution in a GRN presented here, we employ a simple three-state model, as shown in Figure 1. Consider the probabilities ($p_{\mathrm{C,F}}$ and $p_{\mathrm{F,C}}$) of a linage's bidirectional movements from being **c**lose to the optimum to being **f**ar from the optimum (**C** → **F**) and vice versa (**F** → **C**). We expect both to be fairly high in absence of substantial selection for the optimal phenotype. Recombination in particular is a strong force that can substantially alter gene regulation in offspring networks. Therefore, the state transition probabilities for asexual populations may differ from those of sexual populations because mutation has a weaker effect. This also indicates that one of the observations made in Siegal and Bergman (2002) (their Figure 2) may not be correct for asexual linages since the authors only reported the results in sexual linages. In addition, the arguments for the benefits of sex and recombination in Azevedo et al. (2006) and Lohaus et al. (2010) are also incomplete, since these only show $p_{\mathrm{C,U}}$ becomes smaller due to a greatly increased mutational and recombinational robustness, leaving it still unclear whether or not recombination is able to retain sexual lineages in **C**, since $p_{\mathrm{C,F}}$ and $p_{\mathrm{F,C}}$ are largely unknown.

Here we report the discovery of a new possible explanation for the widespread existence of sexual rather than asexual lineages. We found this by exploring systematically the approach to optima with only negligible selection ("no selection" as per Azevedo et al. (2006)) for a particular phenotype, but only when there is selection for developmental stability. Using evolutionary simulations under the GRN model, we show that it is the evolutionary force of recombination together with developmental stability selection that drives populations towards the optimum. We further develop mathematical expressions for the conditions under which this process can be maintained. We find quite surprisingly that recombination does not frequently disrupt well-adapted lineages as conventionally expected. Rather it facilitates finding good genetic combinations robust to disruption, though it also rapidly disrupts weaker configurations. Our results indicate a fundamental difference between recombination and hyper-mutation, which has important implications for the role of gene regulation in the evolution of sex, and for the use of structured representations in machine learning.

Methods

Genotype & Phenotype

In this paper, we use the gene regulatory network (GRN) model originally proposed by Wagner (1994, 1996), and developed by Siegal and Bergman (2002). In the GRN model, the genotype is presented as a network which contains interactions among transcriptional genes. Formally, for each individual network in a finite population M, an $N \times N$ matrix W is an artificial gene network that contains the regulatory interactions among N genes (see Figure 2A). Each element $w_{i,j}$ $(i, j = 1, 2, \ldots, N)$ represents the regulatory effect on the expression of gene i of the product of gene j (see Figure 2B). The network connectivity parameter c determines the proportion of non-zero elements in the network W. A zero entry means there is no interaction between two genes. Through gene interactions, the regulatory effect acts on each gene expression pattern. The phenotype for a given network W is denoted by a state vector $\mathbf{S}(t) = (s_1(t), s_2(t), \ldots, s_N(t))$ where $s_i(t)$ represents the expression level of gene i at time t. Each value of expression state $s_i(t)$ is within the interval $[-1, +1]$ that expresses complete repression (-1) and complete activation $(+1)$. The basic idea is that the individual's phenotype is controlled by the complex interactions between its genes. Formally, for a given gene regulatory network W, the dynamics of $\mathbf{S}$ for each gene i is modelled by

$$s_i(t+1) = f\left(\sum_{j=1}^{N} w_{ij} s_j(t)\right), \qquad (1)$$

where $f(x)$ is a sigmoidal function. In this paper, we define $f(x) = 2/(1+e^{-ax}) - 1$ (Siegal and Bergman, 2002), where a is a free parameter determining the rate of change from complete repression to complete activation. When a is large enough, for example $a = 100$, $f(x)$ is similar as a step function where $f(x) = -1$ for $x < 0$, $f(x) = +1$ for $x > 0$ and $f(0) = 0$.

In all simulations, we define a network's developmental stability as the progression from an initial expression state to an equilibrium expression state (reaching a fixed pattern) by iterating Eq. (1) a fixed number of times, $devT$. If a given network W can achieve developmental stability, then it is termed a viable or stable network, otherwise it is labelled unviable or unstable. We consider an equilibrium state to be reached when the following equation is met:

$$\frac{1}{\tau}\sum_{\theta=t-\tau}^{t} D\left(\mathbf{S}(\theta), \overline{\mathbf{S}}(t)\right) \leq 10^{-4}, \qquad (2)$$

where $D(\mathbf{S}, \overline{\mathbf{S}}) = \sum_{i=1}^{N} (s_i - {s'}_i) \Big/ 4N$ measures the difference between the gene expression pattern $\mathbf{S}$ and $\overline{\mathbf{S}}$, and $\overline{\mathbf{S}}$ is the average of the gene expression levels over the time interval $[t-\tau, t-\tau+1, \ldots, t]$. We use $N = 10$, $c = 0.75$, $devT = 100$, $\tau = 10$ and $a = 100$ in all simulations.

Fitness Evaluation

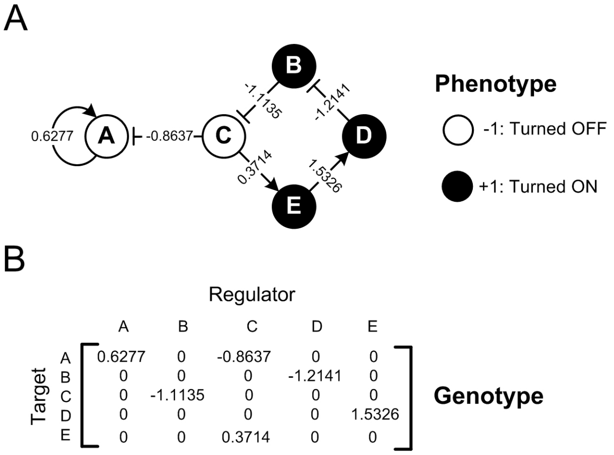
For networks that achieve developmental stability (reaching an equilibrium state, $\mathbf{S}_{\mathrm{EQ}}$), we then calculate fitness as (Siegal and Bergman, 2002)

$$F(\mathbf{S}_{\mathrm{EQ}}) = \exp\left(-\frac{D(\mathbf{S}_{\mathrm{EQ}}, \mathbf{S}_{\mathrm{OPT}})}{\sigma}\right), \qquad (3)$$

where σ is the selection pressure that we imposed on the population during evolution. $\mathbf{S}_{\mathrm{OPT}}$ is usually set to $\mathbf{S}(0)$. $D(\mathbf{S}_{\mathrm{EQ}}, \mathbf{S}_{\mathrm{OPT}})$ is the phenotypic distance between the equilibrium state and the optimal state. Unless otherwise specified, we assign zero fitness to individuals that cannot reach developmental equilibrium.

Initialisation

The initial population contains $M = 10,000$ identical clones of a founder network, which is generated by randomly filling W with $c \cdot N^2$ non-zero elements $w_{i,j}$ drawn from the standard normal distribution, $N(0, 1)$. The associated initial expression state $\mathbf{S}(0)$ is setting each $s_i(0)$ to $+1$. We require that the founder network is not only able to achieve developmental stability but also able to express the optimal state, i.e., $D(\mathbf{S}_{\mathrm{EQ}}, \mathbf{S}(0)) = 0$.



	A	B	C	D	E
A	0.6277	0	-0.8637	0	0
B	0	0	0	-1.2141	0
C	0	-1.1135	0	0	0
D	0	0	0	0	1.5326
E	0	0	0.3714	0	0

Figure 2: An example of gene regulatory network. (A), Network representation of the regulatory iteration between five genes. Open and filled circles represent the genes that are completely turned OFF (-1) or ON $(+1)$, respectively. The initial gene expression pattern is $\mathbf{S}(0) = (-1, +1, -1, +1, +1)$. This example network is stable as it can reach an equilibrium pattern, which is $\mathbf{S}_{\mathrm{EQ}} = (-1, -1, +1, +1, +1)$ by iterating Eq. (1). (B), Interaction matrix (W) represents the network in (A). Each element in row i and column j, i.e., w_{ij}, represents the regulatory effect on the expression of gene i of the product of gene j.

Mutation

For an individual network, each non-zero entry in the W interaction matrix is replaced by ${w'}_{i,j} \sim N(0, 1)$ with mutation rate μ. The expected number of mutations in W is drawn from the Poisson distribution with expectation parameter $\lambda = \mu$. In all simulations, we used $\mu = 0.1$, which means on average there is 0.1 non-zero entry in W that will be mutated per network per generation. Note that mutations should be viewed as operating on the $c \cdot N^2$ *cis*-regulatory elements, not the coding sequences of the N genes themselves (Wagner, 1996; Siegal and Bergman, 2002). In other words, the mutation operation cannot change the topology of the original network W.

Recombination

A recombinant is produced by picking two individuals and selecting rows of the W matrices from each parent with equal probability. This process is similar to free recombination between units formed by each gene and its *cis*-regulatory elements, but with no recombination within regulatory regions.

Stability & Fitness Selection

If individuals are evolved under a regime of stability selection, we only allow ones that can reach developmental equilibrium to stay in the population. Otherwise, if individuals are evolved without stability selection, then we allow both stable and unstable individuals in the population. In all simulations, we set $\sigma = 10^9$, which means all populations are evolved under extremely weak or absent selection for the optimal phenotypic state.

Perturbation Test

In order to measure the distance within the population, or the distance between the population and the optimal phenotype, we use a similar perturbation test as described in Siegal and Bergman (2002): We define a perturbation as a single mutant, i.e., exactly one non-zero entry in W is replaced by a random value drawn from $N(0, 1)$. For each individual in the population, if its perturbed network can still reach a steady state, we define it as $\mathbf{S}_P$, otherwise $\mathbf{S}_P = \bar{\mathbf{S}}(devT)$. The distances between the perturbed individual and its unperturbed one or the optimal phenotypic state are defined as $D(\mathbf{S}_P, \mathbf{S}_{\mathrm{EQ}})$ and $D(\mathbf{S}_P, \mathbf{S}_{\mathrm{OPT}})$, respectively. For each individual, the distance is estimated by averaging 10 perturbations. The reported results are averaged over $10,000$ individuals in the population, total $100,000$ perturbations.

State Transition Probability Measurement

To measure all state transition probabilities p_{ij} $(i, j \in \{\mathbf{C}, \mathbf{F}$ and $\mathbf{U}\})$ as shown in Figure 1, we need to find two populations that are in state $\mathbf{C}$ and state $\mathbf{F}$ for both asexual and sexual populations, respectively. Specifically, as Figure 3 indicates, we use the below four evolved populations for further analysis purpose: For individuals that are close to the optimum, we choose the asexual and sexual populations that have been evolved 1000 generations with stability selection; For individuals that are far away from the optimum, we choose the stable individuals from the asexual and sexual populations that have been evolved 1000 generations without stability selection. Note that the chosen sexual and asexual populations are not perfect but reasonable approximations of two sets of individuals that are in state $\mathbf{C}$ and state $\mathbf{F}$, respectively.

For these four populations, each individual will experience either asexual or sexual reproduction, and be subjected to one single mutation (replacing exactly one non-zero w_{ij} with a random value drawn from $N(0, 1)$). We can easily take the proportion of stable offspring as $1 - p_{\mathrm{C,U}}$ and $1 - p_{\mathrm{F,U}}$. Those stable mutants are saved to further measure the remaining four parameters in Figure 1. We take the mean and standard deviation of the phenotype distance to the optimum at the 1000^{th} generation from sexual population (solid red line in Figure 3) as a criterion for $\mathbf{C}$ to test whether the mutants are closer to the optimum. Similarly, we take the mean and standard deviation of the phenotype distance away from the optimum at the 1000^{th} generation from asexual population (dashed blue line in Figure 3) as a criterion for $\mathbf{F}$ to test whether the mutants are further away from the optimum.[1] For each mutant that is derived from $\mathbf{C}$, if its phenotypic distance is smaller than 2 standard deviations, we count it as in $p_{\mathrm{C,C}}$, otherwise it will be counted as in $p_{\mathrm{C,F}}$. Similarly, for each mutant that is derived from $\mathbf{F}$, if its phenotypic distance is greater than 2 standard deviations, we count it as in $p_{\mathrm{F,F}}$, otherwise it will be counted as in $p_{\mathrm{F,C}}$.

Evolution

As in previous studies (Wagner, 1996; Siegal and Bergman, 2002; Azevedo et al., 2006; Lohaus et al., 2010), the evolutionary simulations are performed under the reproduction-mutation-selection life cycle. The population size M is fixed in every generation throughout evolution in all simulations. In typical asexual evolution, an individual is chosen at random to reproduce asexually by cloning itself, and then subject to mutation, then extremely weak selection for the optimal phenotype. Similarly, in typical sexual evolution, two individuals are chosen at random to reproduce sexually by recombining two parent networks, and then subject to mutation, then extremely weak selection for the optimal phenotype. This process is repeated until the number of M networks are produced. Depending on whether or not the population evolves under a stability selection regime, we exclude unstable networks or allow these networks to stay in the population, accordingly.

Results

Evolution of Phenotypic Distance

We design two sets of simulations to investigate how mutation and recombination influence evolutionary dynamics in asexual and sexual populations using artificial gene regulatory networks. Unlike the simulation set-ups in Siegal and Bergman (2002) and Azevedo et al. (2006), we have taken both sexuality and developmental stability into consideration. For each set of experiments, the results are presented for simulations under four different evolutionary scenarios: 1) population with asexual reproduction and stability selection, 2) population with asexual reproduction but without stability selection, 3) population with sexual reproduction and stability selection, and 4) population with sexual reproduction but without stability selection. In all four cases, the selection for the optimal phenotype is extremely weak or even absent. The reported results are the mean values averaged over 10 replicated simulations, using different randomly generated founder networks.

In the first set of experiments, we measure the phenotypic distance between the evolved populations and the optimum. We compare the results of asexual and sexual populations

[1] Since asexual and sexual populations behave similarly (cf. two dashed lines in Figure 3) when there is no selection for developmental stability, it does not matter whether we take the mean and standard deviation of the phenotype distance from asexual population or sexual population as a criterion for $\mathbf{F}$.

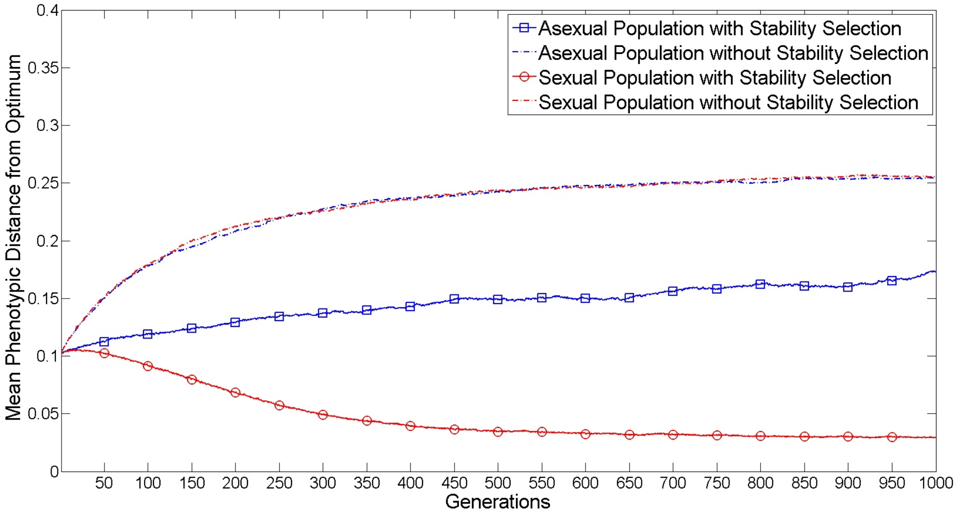


Figure 3: Comparison of phenotypic distance between the evolved populations and the optimum. The asexual (blue lines) and sexual (red lines) populations were evolved 1000 generations under a stability selection (solid lines) or no stability selection (dashed lines) regime, respectively. Key parameters: $M = 10,000$, $N = 10$, $c = 0.75$, $\mu = 0.1$ and $\sigma = 10^9$

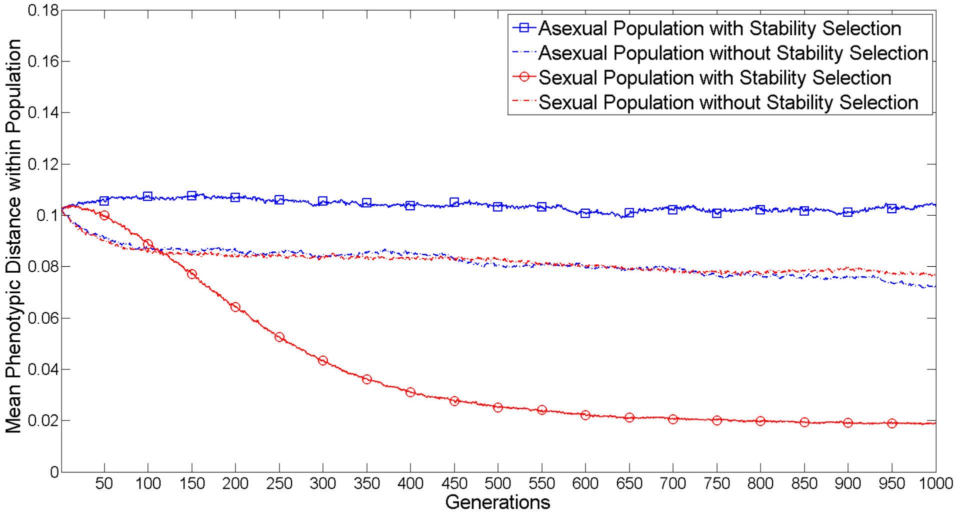


Figure 4: Comparison of phenotypic distance within the evolved populations. The asexual (blue lines) and sexual (red lines) populations were evolved 1000 generations under a stability selection (solid lines) or no stability selection (dashed lines) regime, respectively. Key parameters: $M = 10,000$, $N = 10$, $c = 0.75$, $\mu = 0.1$ and $\sigma = 10^9$

under stability or no stability selection regimes as shown in Figure 3. When there is no stability selection, both asexual and sexual populations (dashed lines in Figure 3) rapidly move away from the optimum at a similar increasing rate. In contrast, when stability selection is imposed on the sexual population (solid red line in Figure 3), the phenotypic distance is continuously decreasing. Although under a stability selection regime the phenotypic distance of asexual population (solid blue line in Figure 3) still increases slightly, stability selection greatly helps impede deviation, compared with the situation when stability selection is absent. These results suggest that the combination of the two forces of recombination and stability selection can drive the population towards the optimum.

In the second set of experiments, we measure the phenotypic distance within the evolved populations. Similarly, we compare the results of asexual and sexual populations under stability or no stability selection regimes as shown in Figure 4. In contrast to the results where we compare phenotypic distance between the evolved populations and the optimum, the phenotypic distance within the populations reduces when there is no stability selection in both asexual and sexual populations (dashed lines in Figure 4) with a similar modestly decreasing rate. However, there is no obvious change in phenotypic distance within the asexual population (solid blue line in Figure 4) when we include stability selection. But the phenotypic distance of the sexual population (solid red line in Figure 4) is highly reduced. This indicates that recombination and stability selection can also help sexual lineages stay close to each other. Note that this phenomenon has been similarly reported in Siegal and Bergman (2002), except it can be only observed in sexual lineages rather than asexual lineages (cf. solid red and blue lines in Figure 4).

Analysis

From Figures 3–4, we can clearly see that phenotypic distance is continuously decreasing in the sexual population as the consequence of recombination and selection for developmental stability. In order to fully understand how these two forces act together, here we further investigate the underlying evolutionary dynamics in the context of artificial gene regulatory networks.

The evolutionary dynamics in the GRN model can be regarded as a Markov process since the future of the evolution process is based solely on its present state. To simplify the analysis, we define three states in the system as per Figure 1:

- **C**: Individuals that are close to the optimal phenotype.
- **F**: Individuals that are far from the optimal phenotype.
- **U**: Individuals that are unable to achieve developmental stability, and thus will be removed from the population.

Suppose at g^{th} generation, the frequency of individuals in state C is $f_{\text{C}}(g)$, so the frequency of individuals in state $f_{\text{F}}(g)$ is $f_{\text{F}}(g) = 1 - f_{\text{C}}(g)$. Therefore, the changing rate of frequency $f_{\text{C}}(g)$ between two consecutive generations can be described as $\Delta\text{C}/\Delta g = f_{\text{C}}(g+1) - f_{\text{C}}(g)$. In order to observe a decreased phenotypic distance between the current population and the optimum, the following condition should be met: $\Delta\text{C}/\Delta g \geq 0$. More formally (see Appendix), the below quadratic equation should be satisfied:

$$(p_{\text{F,U}} - p_{\text{C,U}})f^2_{\text{C}}(g) + (2p_{\text{F,C}} + p_{\text{F,F}} - p_{\text{C,C}})f_{\text{C}}(g) - p_{\text{F,C}} \leq 0. \tag{4}$$

It is reasonable to assume that individuals that are far away from the optimum (**F**) are more likely to become unstable than individuals that are close to the optimum (**C**), i.e., $p_{\text{F,U}} \geq p_{\text{C,U}}$. This is because there is always a selection for developmental stability that enables population moving towards the optimum, whereas the stability selection is absent, and consequently drags the population moving away from the optimum. Given that $f_{\text{C}}(g) \in [0, 1]$ and $p_{\text{F,C}} \geq 0$, in order to let Eq. (4) hold, $p_{\text{C,F}} \approx 0$ should hold. This suggests that as long as the population is continuously moving towards the optimum, the evolved lineages are unlikely deviated by mutation and recombination from the area close to the optimum (**C**) to the area far from it (**F**).

From the above analysis, we can speculate that as long as $p_{\text{C,F}}$ is smaller enough, we should be able to see an increased frequency of lineages in **C** state. From the observation of our evolutionary simulations, we further expect that $p_{\text{C,F}}$ should be smaller in sexual lineages than in asexual lineages since sexual population is moving more quickly towards the optimum, whereas a similar pattern has not been observed in asexual population.

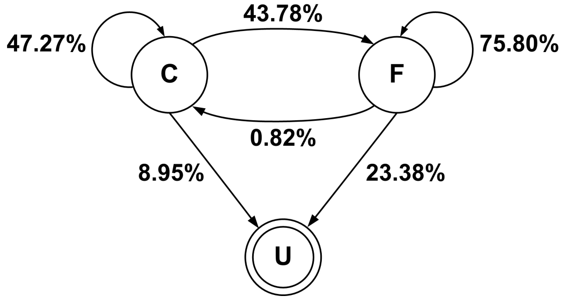


Figure 5: Mean estimated state transition probabilities in asexual populations. $p_{\text{C,U}}$: 8.95% (SD: 0.28%), $p_{\text{C,F}}$: 43.78% (SD: 0.44%), $p_{\text{C,C}}$: 47.27% (SD: 0.43%), $p_{\text{F,U}}$: 23.38% (SD: 0.36%), $p_{\text{F,C}}$: 0.82% (SD: 0.08%), $p_{\text{F,F}}$: 75.80% (SD: 0.38%). SD: Standard Deviation

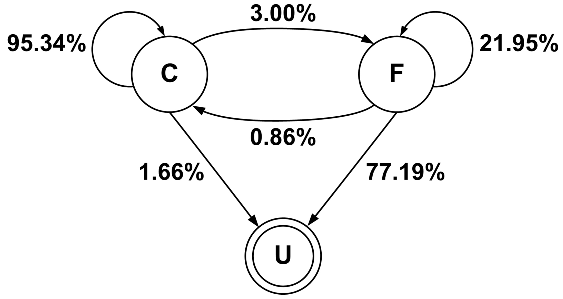


Figure 6: Mean estimated state transition probabilities in sexual populations. $p_{\text{C,U}}$: 1.66% (SD: 0.13%), $p_{\text{C,F}}$: 3.00% (SD: 0.15%), $p_{\text{C,C}}$: 95.34% (SD: 0.18%), $p_{\text{F,U}}$: 77.19% (SD: 0.43%), $p_{\text{F,C}}$: 0.86% (SD: 0.08%), $p_{\text{F,F}}$: 21.95% (SD: 0.42%). SD: Standard Deviation

Numerical Simulations

To verify our analysis, we now treat our original simulations as numerical simulations to measure the state transition probabilities p_{ij} $(i, j \in \{\mathbf{C}, \mathbf{F} \text{ and } \mathbf{U}\})$. We use the four evolved populations described above (see Method). The reported results are based on 50 independent runs as shown in Figures 5–6 for asexual and sexual populations, respectively.

We can clearly see in Figures 5–6 that $p_{\mathrm{F,U}} \geq p_{\mathrm{C,U}}$ holds for both asexual and sexual populations. But only in sexual population, $p_{\mathrm{C,F}}$ is a small value 0.0300^2, whereas $p_{\mathrm{C,F}}$ is a much larger value 0.4378 in asexual population. These results confirm that Eq. 4 holds only for sexual populations when there is no selection for the optimal phenotype. But please note that this does not suggest that Eq. 4 will never hold in asexual populations. When selection for the optimal phenotype is turned on, we can also observe a small value of $p_{\mathrm{C,F}}$ in asexual population (results not shown).

From Figures 5–6, we can further calculate the stationary probabilities of lineages with **C** and **F** states for asexual and sexual populations. For the asexual population: 2.68% in **C** area and 97.32% in **F** area. For sexual population: 96.09% in **C** area and 3.91% in **F** area. These results demonstrate that recombination substantially enables sexual lineages to sustain themselves near an optimum to a surprisingly high probability. This further indicates a fundamental difference between recombination and hyper-mutation, despite their superficial similarity in causing increased variation.

Discussion & Conclusion

Mutation and recombination are two important evolutionary forces that provide heritable genetic innovations that ultimately stimulate adaptation for species to survive in nature. But compared with mutation, recombination is thought to be much more mysterious because it leads to a fundamental evolutionary question: how sexual reproduction, once evolved, can be maintained in the long term. In particular, we do not clearly understand why asexual lineages do not outcompete sexual lineages, given the substantial cost of recombination (disrupting good genetic combinations) and the two-fold cost of sex (producing half as many lineages because only females reproduce). Although previous studies have posited that recombination has an important role in improving robustness and facilitating evolutionary adaptation, the underlying mechanism has been unclear.

Although Siegal and Bergman (2002) have emphasised the importance of the "phenotypic buffer" for genetic innovation provided by developmental stability selection, they did not take sexuality into consideration. In later studies (Azevedo et al., 2006; Lohaus et al., 2010), the authors have reported the observed mutational and recombinational robustness evolved from sexual lineages, but they still did not clearly explain the great benefit observed in sexual populations. Here, we have used a simple, three-state system to describe the evolutionary process in the GRN model (Figure 1). With this we have shown that even if both the mutational and recombinational robustness are greatly increased (i.e. $p_{\mathrm{C,U}}$ is reduced), the underlying dynamics of sexual lineages have been poorly understood. One reasonable intuition would be a high transition probability of lineages moving from area **C** to **F** since recombination is thought to greatly disrupt well-adapted lineages, and consequently it would be thought of as unlikely to sustain population **C**, close to the optimum. Therefore $p_{\mathrm{C,F}}$ would be expected to be high. Instead, we find that while $p_{\mathrm{F,U}}$ is high as similarly predicted, individuals of sexual lineages that are close to the optimum appear to be very robust against disruption by recombination. We take this as clear evidence of evolvability.

By comparing evolutionary simulations of asexual and sexual lineages under developmental stability selection or in its absence, we have shown that the conclusion in Siegal and Bergman (2002) is not complete. It is the combination of two evolutionary forces — recombination and developmental stability selection — that drives populations towards the optimum, not just stability selection alone (Figures 3–4). We have thus clarified the benefit derived from sexual lineages. Our numerical analysis has verified our assumptions and further validated our results. We have shown that recombination is surprisingly constructive for close-to-optimum sexual lineages, and only destructive to populations far from the optimum. This indicates that recombination facilitates the evolution of evolutionary robustness a form of evolvability in sexual lineages. In contrast, mutations are more likely to be universally mildly deleterious and thus after accumulation in an asexual population, ultimately tend to move it away from the optimum. These findings indicate a fundamental difference between recombination and hyper-mutation, which has important implications both for structuring machine learning, and for finally explaining the evolutionary stability of sex.

References

Azevedo, R. B. R., Lohaus, R., Srinivasan, S., Dang, K. K., and Burch, C. L. (2006). Sexual reproduction selects for robustness and negative epistasis in artificial gene networks. *Nature*, 440(7080):87–90.

Barton, N. H. (2009). Why sex and recombination? *Cold Spring Harbor Symposia on Quantitative Biology*, 74(0):187–195.

Eshel, I. and Feldman, M. W. (1970). On the evolutionary effect of recombination. *Theoretical Population Biology*, 1(1):88–100.

Feldman, M. W., Otto, S. P., and Christiansen, F. B. (1996). Population genetic perspectives on the evolution of recombination. *Annual Review of Genetics*, 30:261–295.

Hu, T., Banzhaf, W., and Moore, J. H. (2014). The effects of recombination on phenotypic exploration and robustness in evolution. *Artificial Life*, 20(4):457–470.

[2] Note that the boundary between $p_{\mathrm{C,C}}$ and $p_{\mathrm{C,F}}$, and the boundary between $p_{\mathrm{F,F}}$ and $p_{\mathrm{F,C}}$ are arbitrarily defined. If we slightly increase the range from 2 standard deviations to 3 standard deviations, then both $p_{\mathrm{C,F}}$ and $p_{\mathrm{F,C}}$ are almost 0. It should be noted that the observation that sexual population's $p_{\mathrm{C,F}}$ are smaller than asexual population's $p_{\mathrm{C,F}}$ holds for any arbitrarily defined boundary (results not shown).

Hurst, L. D. and Peck, J. R. (1996). Recent advances in understanding of the evolution and maintenance of sex. *Trends in Ecology & Evolution*, 11(2):46–52.

Kondrashov, A. S. (1993). Classification of hypotheses on the advantage of amphimixis. *Journal of Heredity*, 84(5):372–387.

Kouyos, R. D., Silander, O. K., and Bonhoeffer, S. (2007). Epistasis between deleterious mutations and the evolution of recombination. *Trends in Ecology & Evolution*, 22(6):308–315.

Le Cunff, Y. and Pakdaman, K. (2014). Reproduction cost reduces demographic stochasticity and enhances inter-individual compatibility. *Journal of Theoretical Biology*, 360:263–270.

Lohaus, R., Burch, C. L., and Azevedo, R. B. R. (2010). Genetic architecture and the evolution of sex. *The Journal of heredity*, 101(Supplement 1):S142–S157.

MacCarthy, T. and Bergman, A. (2007). Coevolution of robustness, epistasis, and recombination favors asexual reproduction. *Proceedings of the National Academy of Sciences*, 104(31):12801–12806.

Martin, O. C. and Wagner, A. (2009). Effects of recombination on complex regulatory circuits. *Genetics*, 183(2):673–684.

Meirmans, S. and Strand, R. (2010). Why are there so many theories for sex, and what do we do with them? *Journal of Heredity*, 101(Supplement 1):S3–S12.

Otto, S. P. and Feldman, M. W. (1997). Deleterious mutations, variable epistatic interactions, and the evolution of recombination. *Theoretical Population Biology*, 51(2):134–147.

Otto, S. P. and Gerstein, A. C. (2006). Why have sex? The population genetics of sex and recombination. *Biochemical Society Transactions*, 34(4):519–522.

Otto, S. P. and Lenormand, T. (2002). Resolving the paradox of sex and recombination. *Nature Reviews Genetics*, 3(4):252–261.

Payne, J. L., Moore, J. H., and Wagner, A. (2014). Robustness, Evolvability, and the Logic of Genetic Regulation. *Artificial Life*, 20(1):111–126.

Siegal, M. L. and Bergman, A. (2002). Waddington's canalization revisited: Developmental stability and evolution. *Proceedings of the National Academy of Sciences*, 99(16):10528–10532.

Wagner, A. (1994). Evolution of gene networks by gene duplications: A mathematical model and its implications on genome organization. *Proceedings of the National Academy of Sciences*, 91(10):4387–4391.

Wagner, A. (1996). Does evolutionary plasticity evolve? *Evolution*, 50(3):1008–1023.

Wagner, A. (2011). The low cost of recombination in creating novel phenotypes. *BioEssays*, 33(8):636–646.

Wang, Y., Matthews, S. G., and Bryson, J. J. (2014). Evolving evolvability in the context of environmental change: A gene regulatory network (GRN) approach. In *Artificial Life 14: International Conference on the Synthesis and Simulation of Living Systems*, pages 47–53. The MIT Press.

West, S. A., Lively, C. M., and Read, A. F. (1999). A pluralist approach to sex and recombination. *Journal of Evolutionary Biology*, 12(6):1003–1012.

Appendix

The evolution dynamics in the GRN model can be regarded as a Markov process. As described in the main text, we simply define three states in the system: **C** (lineages are close to the optimum), **F** (lineages are far from the optimum), **U** (lineages are unstable, and thus unviable). Therefore, suppose at g^{th} generation, the frequency of individuals with state **C** is $f_{\mathrm{C}}(g)$, the frequency of individuals with state **F** is $f_{\mathrm{F}}(g) = 1 - f_{\mathrm{C}}(g)$. Then at $(g+1)^{th}$ generation, the frequency of individuals with state **C** and **F** are $f_{\mathrm{C}}(g{+}1) = (f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}})/k$ and $f_{\mathrm{F}}(g{+}1) = (f_{\mathrm{F}}(g) \times p_{\mathrm{F,F}} + f_{\mathrm{C}}(g) \times p_{\mathrm{C,F}})/k$, respectively, where a normalising factor $k = (f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}}) + (f_{\mathrm{C}}(g) \times p_{\mathrm{C,F}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,F}})$. Therefore, the changing rate of frequency f_{C} between two consecutive generations can be described as in the following differential equation:

$$
\begin{aligned}
\frac{\Delta \mathrm{C}}{\Delta g} &= f_{\mathrm{C}}(g+1) - f_{\mathrm{C}}(g) \\
&= \frac{f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}}}{k} - f_{\mathrm{C}}(g) \\
&= \frac{f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} - k \times f_{\mathrm{C}}(g)}{k}.
\end{aligned}
$$

As population evolves towards the optimum, we expect to see a higher frequency of f_{C} in the population, i.e., $\Delta\mathrm{C}/\Delta g \geq 0$. Therefore, the below equation should be satisfied:

$$
\begin{aligned}
& f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} - k \times f_{\mathrm{C}}(g) \\
&= f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} - f_{\mathrm{C}}(g) \times (f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} \\
&\quad + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} + f_{\mathrm{C}}(g) \times p_{\mathrm{C,F}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,F}}) \\
&= f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} \\
&\quad - f_{\mathrm{C}}(g) \times f_{\mathrm{F}}(g) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,F}} \\
&\quad - f_{\mathrm{C}}(g) \times f_{\mathrm{F}}(g) \times p_{\mathrm{F,F}} \\
&= f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + (1 - f_{\mathrm{C}}(g)) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} \\
&\quad - f_{\mathrm{C}}(g) \times (1 - f_{\mathrm{C}}(g)) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,F}} \\
&\quad - f_{\mathrm{C}}(g) \times (1 - f_{\mathrm{C}}(g)) \times p_{\mathrm{F,F}} \\
&= f_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} + p_{\mathrm{F,C}} - f_{\mathrm{C}}(g) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,C}} \\
&\quad - f_{\mathrm{C}}(g) \times p_{\mathrm{F,C}} + f^2_{\mathrm{C}}(g) \times p_{\mathrm{F,C}} - f^2_{\mathrm{C}}(g) \times p_{\mathrm{C,F}} \\
&\quad - f_{\mathrm{C}}(g) \times p_{\mathrm{F,F}} + f^2_{\mathrm{C}}(g) \times p_{\mathrm{F,F}} \\
&= (p_{\mathrm{F,F}} + p_{\mathrm{F,C}} - p_{\mathrm{C,C}} - p_{\mathrm{C,F}}) \times f^2_{\mathrm{C}}(g) \\
&\quad + (p_{\mathrm{C,C}} - 2p_{\mathrm{F,C}} - p_{\mathrm{F,F}}) \times f_{\mathrm{C}}(g) + p_{\mathrm{F,C}} \\
&\geq 0.
\end{aligned}
$$

Therefore, the following condition should hold:

$$
(p_{\mathrm{F,U}} - p_{\mathrm{C,U}}) \times f^2_{\mathrm{C}}(g) + (2p_{\mathrm{F,C}} + p_{\mathrm{F,F}} - p_{\mathrm{C,C}}) \times f_{\mathrm{C}}(g) - p_{\mathrm{F,C}} \leq 0.
$$

It is reasonable to assume that $p_{\mathrm{F,U}} \geq p_{\mathrm{C,U}}$ since there is always a selection for developmental stability when population is moving towards **C**, whereas the stability selection is absent when the population is moving towards **F**. Therefore, the quadratic equation Eq. (4) is concave up. We know that $f_{\mathrm{C}}(g) \in [0, 1]$, therefore Eq. (4) will hold as long as it holds in $[0, 1]$. Therefore, the below two conditions should be satisfied:

$$
\begin{aligned}
-p_{\mathrm{F,C}} &\leq 0 \\
p_{\mathrm{F,U}} - p_{\mathrm{C,U}} + 2p_{\mathrm{F,C}} + p_{\mathrm{F,F}} - p_{\mathrm{C,C}} - p_{\mathrm{F,C}} &\leq 0
\end{aligned}
$$

Clearly, $p_{\mathrm{F,C}} \geq 0$ always holds, therefore the second condition which is $p_{\mathrm{C,F}} \leq 0$ should hold. But we know that $p_{\mathrm{C,F}} \geq 0$. Therefore, if Eq. (4) holds, then $p_{\mathrm{C,F}} \approx 0$ should hold.

Searching for non-regular neighborhood cellular automata rules applied to scheduling task and guided by a forecast dynamical behavior parameter

Tiago I. Carvalho [1] and Gina M. B. Oliveira [1]

[1]Universidade Federal de Uberlândia
gina@facom.ufu.br

Abstract

Cellular automata (CA) have been recently considered for the scheduling task. Since the problem of forecasting dynamic behavior of CA is undecidable, several parameter-based approximations have been developed to the problem. Sensitivity parameter is one of the most-studied CA forecast parameters and it has shown efficient to identify the dynamical behaviour of CA rules characterized by regular neighborhood. Here we perform an investigation about the usage of sensitivity to identify dynamical classes in cellular automata rules with a non-linear neighborhood model in a scheduling task. The results show that sensitivity can help the identification of long-cycle rules, avoiding this undesirable behavior in the search for CA rules appropriate to schedule the tasks of parallel programs in a multiprocessor architecture.

Introduction

Cellular automata (CA) are discrete dynamical systems composed by simple components with local interactions. The direct way to analyze the dynamics of a cellular automaton is to observe its behavior, through the spatial-temporal patterns it generates, out of several random initial conditions, and then to use metrics to quantify the observed behavior. It is not possible to forecast their dynamic behavior they will perform when running. However, it is possible to derive static parameters from their definition, which can then be used to estimate their dynamic behavior. One of them named Sensitivity *(μ)* was proposed by Binder and it has shown to be a good indicator of chaotic rules (Binder, 1988). However, the forecast parameter applications in the literature were on standard cellular automata models in which the neighborhood is defined by an identical pattern of local connections to other cells. Here we are interested in non-regular neighborhood structures in which the connections are defined based on a graph. More specifically, in a neighborhood model that has been recently used in the CA-based task scheduling problem (Seredynski et al. 2002), (Swiecicka et al., 2006), (Oliveira and Vidica, 2012), (Carneiro and Oliveira, 2013).

The neighborhood structure studied here is named pseudo-linear and it was proposed in (Carneiro and Oliveira, 2013). This neighborhood model has a non-linear structure defined by a graph used to represent the dependence relations between the tasks in a parallel program. Such kind of model was used to solve scheduling task problem in a CA model named SCAS-HP (Synchronous Cellular Automata-based Scheduler initialized by Heuristic and modeled by a Pseudo-linear neighborhood) in (Carneiro and Oliveira, 2013). Although SCAS-HP had returned good results in the tests performed it was later identified that a high number of chaotic rules were generated by the search employed in that model. This characteristic is not desirable since chaotic rules just define random task allocations. Other undesirable behavior identified is represented by complex and periodic rules with a long cycle (above 5 as pointed in (Carneiro and Oliveira, 2013)). In this sense, chaotic, complex and periodic rules with cycle above 5 were grouped here as long-cycle rules; this group represents the dynamical behavior that we want to avoid in scheduling applications.

First, we performed some analyses in which it was possible to conclude that the sensitivity parameter is good to characterize the long-cycle behavior in the space of rules with pseudo-linear neighborhood. Subsequently, we investigated if we can use sensitivity to guide the genetic search performed in SCAS-HP to find good scheduler rules with also a desirable dynamical behavior, avoiding long-cycle rules. Using this parameter-based information in the genetic search, we proposed a new CA-based scheduler named here as SCAS-HPμ (Synchronous Cellular Automata-based Scheduler using Heuristic initialization, Pseudo-linear neighborhood and guided by μ parameter).

Cellular Automata

Considering its standard definition, a cellular automaton has a regular lattice of *N* cells, each one with an identical pattern of local connections to other cells. CA transition rule determines the state of the cell *i* at time $t + 1$ depending on the states of its neighborhood at time *t* including cell *i*. The state of cell *i*, together with the states of all cells connected with *i* are called the neighborhood of cell *i*. Typically, a one dimensional CA defines its neighborhood using a radius specification. For example, elementary CA has 1D lattice, 2 states and radius 1 neighborhood formed by three cells: the cell itself and its immediate neighbors in the lattice (to the right and to the left). The transition rule defines the output state for each possible neighborhood, that is, the new state of the central cell for each possible configuration of its current neighbor states.

The transition rule is defined by 8 output bits in the elementary CA (one bit for each one of the eight 3-cells binary neighborhood). Elementary cellular automaton is the simplest CA model and its complexity can be improved in several ways. One can use a two-dimensional lattice instead of the 1D (or any k-dimensional with $k > 2$). Even using the 1D lattice, one can use a larger radius to define a more connected neighborhood; for example, using radius 2 the neighborhood has 5 cells. Another possible variation is to increase the number of states of the model. Here we are interested in investigating a non-standard model of cellular automata in which the neighborhood is based on a non-regular pattern of connections. They are based on neighborhoods in which each cell can be connected with other cells not necessarily in the vicinity of its lattice position. The model discussed here was named pseudo-linear neighborhood in (Carneiro and Oliveira, 2013) in which the neighbors of each cell is extracted by non-linear relations in a graph.

CA-based task scheduling

Task scheduling aims to allocate a set of computational tasks that compose a parallel application in the nodes of a multiprocessor architecture. Considering Task Static Scheduling Problem (TSSP), all information about the tasks is known a priori. An optimal solution for an instance of TSSP is such that the precedence constraints are satisfied and the runtime - or makespan - is minimized. The problem is NP-Complete, and it is a challenge for many researchers. The proposed approaches to solve it typically employ heuristics or meta-heuristics. However, such heuristics do not have the ability to extract knowledge of the scheduling process of a parallel application and reuse it in other instances. Previous works pointed to the promising use of CA-based approaches to TSSP (Seredynski, et al. 2002), (Swiecicka et al. 2006), (Oliveira and Vidica, 2012), (Carneiro and Oliveira, 2013). Such models combine the use of CA and Genetic Algorithms (GA) due to the employment of transition rule spaces with high cardinality (Mitchell, 1996).

A parallel application is represented by a program graph which is a weighted directed acyclic graph $G = (V, E)$. V is the set of N tasks of the parallel program. It is assumed that each task is a computational indivisible unit. There is a precedence constraint relation between tasks i and j if the result produced by task i has to be available to task j, before it starts. E is the set of precedence relations between tasks. A program graph has weights associate to nodes and edges: b_k describes the processing time required to execute a given task on any processor and a_{kl} describes the communication time between pairs of tasks k and l, when they are located in distinct processors. The purpose of scheduling is to distribute the tasks of a parallel program among the processors in such a way that the precedence constraints are preserved and the total execution time (or makespan) T is minimized. Figure 1 shows an example of a program graph with 18 tasks named *Gauss18* (Cosnard et al., 1988).

A CA-based scheduler was presented in (Seredynski et al., 2002). It operates in two phases: learning and operation. In the learning mode, a genetic algorithm (GA) (De Jong, 2006) is used to search for rules able to converge the lattice to optimal (or sub-optimal) allocations of a given program graph, starting from random initial configurations. In the operation mode it is expected that, for any initial allocation of tasks, the CA rules stored after learning are able to evolve the lattice until a configuration which represents an optimal allocation, that is, it minimizes the makespan. It is also expected that the rules obtained in the learning phase can be used in the scheduling of other graphs (reuse capability). The neighborhood model employed in (Seredynski et al., 2002) is the regular one employed on standard CA and it was called as linear neighborhood by the authors. Both updating modes of cells - sequential and parallel – were investigated being that the results obtained with the sequential updating were better.

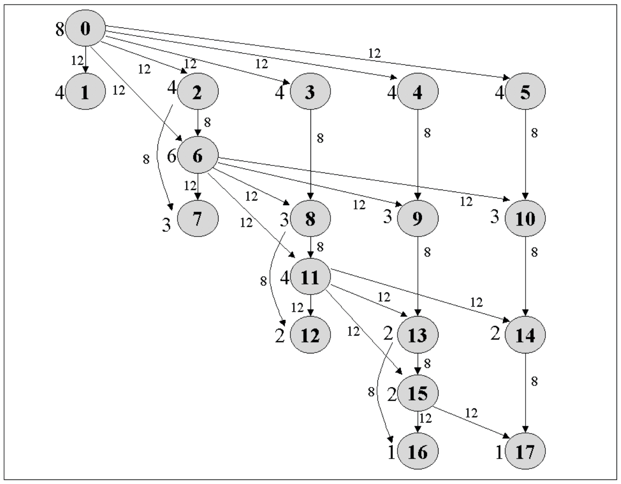


Fig.1: Program Graph *Gauss18*.

In the initial CA-based scheduler models the synchronous updating mode of cells was discarded because it returned the worst results (Seredynski et al., 2002). However, the large capacity of parallelism inherent to CA is lost if the asynchronous updating of cells is adopted. A CA-based scheduler model using synchronous updating of cells was presented in (Carneiro and Oliveira, 2011). This model was named SCAS (Synchronous Cellular Automata-based Scheduler) and it also employs linear neighborhood, but unlike the models in (Seredynski et al., 2002) and (Swiecicka et al., 2006) the strategy in GA is not elitist. In addition, the boundary condition used in CA lattice are different from the null condition employed in (Seredynski et al., 2002) and (Swiecicka et al., 2006): cells to the right of last cell are considered in the state 1. The results obtained using SCAS showed the good performance of this model when compared to the previous synchronous updating models .

Later, a scheduler model based on CA was presented in (Carneiro and Oliveira, 2013) and it was named SCAS-HP (Synchronous Cellular Automata-based Scheduler initialized by Heuristic and modeled by a Pseudo-linear neighborhood). In such model, each lattice cell is associated with a task of the program graph. Considering k-processor architectures, the lattice has k states possible per cell and each cell state indicates that the corresponding task is allocated either in the processor P_0, P_1 or P_k. Each lattice configuration corresponds to a different allocation of the tasks in the processors. Makespam T associated to a given lattice is calculated using a scheduling policy that defines the order of the tasks within each processor.

One important modification of SCAS-HP compared with the previous CA-based schedulers refers to the way the CA lattice is initialized to perform the schedule, which reflects in the process of evaluation of GA rule population (*Pop*). The main steps in evaluation used in SCAS-H are: (i) applying a deterministic construction heuristic chosen a priori to obtain an initial allocation which defines the initial configuration of the lattice; (ii) temporal evolution of the lattice using each rule transition *r* in *Pop* for *T* time steps; (iii) the final lattices in time *T* define the final allocations of tasks which are ordered in each processor using a scheduling policy obtaining makespan associated to each rule *r*; (iv) rule fitness is equal to makespan obtained using it. The best rule in *Pop* presents the smallest makespan. The construction heuristic investigated in (Carneiro and Oliveira, 2013) is the DHLFET (Deterministic Highest Level First with Estimated Time), based on the widely known HLFET heuristic (Adam and Chanky, 1974). Another important characteristic of SCAS-HP model (Carneiro and Oliveira, 2013) is the use of a non-regular neighborhood able to capture the spatial relations of the computational tasks in the program graph. It was named pseudo-linear neighborhood, since it preserves the simple structure of linear (regular) neighborhoods, but the neighbors relations are defined by the proximity and relative importance of the tasks within the program graph. Pseudo-linear considers only the predecessors or successors of a given task. For each task and edge in the graph is associated a cost, which detects the strength of the relationship among one task and its predecessor or successor.

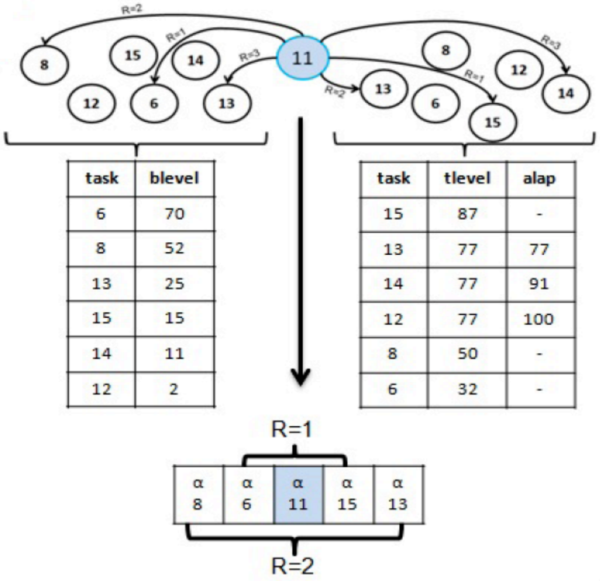
Pseudo-linear neighborhood employs two well-known attributes in task scheduling (Carneiro and Oliveira, 2013): bottom level of a task (or blevel) and top level of a task (or tlevel). Given a radius *R*, the next state of a cell *i* (σ_i) is defined according to the equation:

$$\sigma_i^{\tau+1} = \Delta[\sigma^{\tau}_{blevel(R)}, \ldots, \sigma^{\tau}_{blevel(1)}, \sigma_i^{\tau}, \sigma^{\tau}_{tlevel(1)}, \ldots, \sigma^{\tau}_{tlevel(R)}]$$

Initially, the set of successors and predecessors of task *i* are identified in a list of related tasks. Thus, the list of tasks is ordered by *blevel* attribute in a descending order and function *blevel(x)* returns the x^{th} task in this ordered list. A similar procedure is performed for function *tlevel(x)*, which returns the x^{th} related task considering the tlevel descending order. For each attribute (*blevel* and *tlevel*), using a neighborhood with radius *R*, only the first *R* tasks in each list are used as neighbors. Note that when two or more tasks have the same value for some attribute, *blevel* orders these tasks according to the highest order number, whereas *tlevel* orders these tasks according to the lowest *alap* (As Late As Possible start time). If two or more tasks have the same *alap*, *tlevel* list orders these tasks according to the lowest order number. Furthermore, there is only one special case in Pseudo-linear neighborhood: when the number of neighbors is smaller than *R*. In this case, the lists come back to head task and continue until they complete *R* neighbors tasks.

Figure 2 illustrates the pseudo-linear neighborhood using the example of Task 11 in Gauss18 graph (Fig. 1). Using a radius *R* = 2, it considers tasks 6, 8, 13 and 15 as the neighbors of the cell/task 11 because they are the tasks with a direct relation with node 11 (successor or predecessor) with the highest value of *blevel* (nodes 6 and 8) and *tlevel* (nodes 13 and 15) it is able to capture proximity and strength of the relations among tasks in the Gauss18 program graph.

Experiments with SCAS-HP and some previous models performed in (Carneiro and Oliveira, 2013) showed the application of pseudo-linear neighborhood makes the search for CA transition rules during learning more robust, leading to a significant gain when considering the reuse of them on real-world conditions.



task	blevel
6	70
8	52
13	25
15	15
14	11
12	2

task	tlevel	alap
15	87	-
13	77	77
14	77	91
12	77	100
8	50	-
6	32	-

α 8	α 6	α 11	α 15	α 13

Fig.2: Definition of the neighborhood structure of Task 11 from Gauss18, using the pseudo-linear model.

CA Dynamics and Sensitivity Parameter

CA can exhibit a rich variability of dynamic behaviors. Wolfram proposed a qualitative behavior classification in four dynamic classes, which is widely known (Wolfram, 1994). In (Li and Packard, 1990) proposed a refinement which divides the rule space into six classes: Null, Fixed-Point, Two-Cycle, Periodic, Complex and Chaotic. For scheduling application purpose, chaotic dynamics must be avoided; moreover, periodic rules with cycle above 5 and complex rules are also undesirable. The major reason to avoid periodic behaviors with cycle above 5 is that the scheduler model discards the results of this type of rule when it is verifying the makespam associated to a rule (Carneiro and Oliveira, 2015). Therefore, we adopted an approximate scheme which divides the rule space into five classes:

- **Null**: the limiting configuration is only formed by 0s or 1s.
- **Fixed-Point**: the limiting configuration is invariant (with possibly a spatial shift) by applying the cellular automaton rule, the null configurations being excluded.
- **Two-Cycle**: the limiting configuration is invariant, with possibly a spatial shift, by applying the rule.
- **Periodic** (until cycle length 5): the limiting configuration is invariant by applying the automaton rule *L* times (with *L* < 6), with the cycle length *L* either independent or weakly dependent on the system size.
- **Long-Cycle**: collapses two Li and Packard's classes and a subset of a third class into just one class: (i) periodic rules with cycle length bigger than 5; (ii) complex rules and (iii) chaotic rules. Based on the study of regular cellular automata

spaces it is known that the number of chaotic rules is much more than period long cycle rules or complex ones. Thus, this class is essentially formed by chaotic rules. These rules are characterized by the exponential divergence of its cycle length with the system size, and for the unstability with respect to perturbations.

A scheme to do an automatic classification of the dynamics exhibits by a CA rule was elaborated. The automatic classification of each CA rule is done by applying the rule over the lattice starting from the initial configuration for a number of steps to observe its behavior and a sample of ICs was used to identify the average behavior of the rule. Considering the automatic classification scheme adopted, the following steps are applied: (i) random generation of 100 initial lattices of size *N*; (ii) application of the rule transition by *T* time steps (*T* = 3*N* was adopted here) for each initial lattice of the sample; (iii) classification of the observed dynamics in each temporal evolution (iv) computation of the dynamics which appears in the majority of the evolutions. Here we will present a brief review of this automatic classification applied to two sets of 1D CA rules: (i) the set of 256 CA elementary rules using the standard spatially-local neighborhood ; (ii) the set of 256 1D CA rules using the standard CA spatially-local neighborhood.

Considering the elementary rule space (composed by 256 rules with 1D-lattice, binary states and radius 1 neighborhood), a comparison between the classical classification performed by Li and Packard and the classification obtained using the automatic classifier was done. The results show that 205 out of 256 rules did not change their classification; 6 rules originally classified as complex and 32 rules classified as chaotic were classified in the new scheme as long-cycle (as planned). The other 13 rules have changed their classification due to different reasons: some of them were originally classified as periodic because they exhibited a cycle length between 6 and 10 and other are very hard to classify because they are strongly dependent on the random lattice sample employed. Therefore, the automatic classification returned the expected classes in the elementary space and it is very close to the classification performed by Li and Packard.

The dynamics of a cellular automaton is associated to its transition rule, and it has been proved that the correct forecast of its dynamic behaviour, out of its rule, is an undecidable problem. Several parameters directly calculated from the rule table have been proposed to help forecast CA dynamic behavior (Oliveira , et al. 2001).

Binder proposed the sensitivity parameter (μ) (Binder, 1988). Its notion is motivated by the numeric observation that the dynamical classes are characterized by its sensibility to changes in the state of a unique cell of the neighborhood of the CA transition rule. Sensitivity is defined as the number of changes in the outputs of the transition rule, caused by changing the state of a cell of the neighborhood, each cell at a time, over all possible neighborhoods of the rule being considered.

The sensitivity parameter (normalized between 0 and 1) was calculated for all 256 rules of the elementary space and the values were grouped according to the dynamic behavior of each rule. Figure 3 shows the results of such analysis. The null behavior happens in rules with low sensitivity (until 0.5) and the long-cycle behavior happens in rules with high sensitivity (starting from 0.5). Fixed-point and periodic (short cycle) behaviors are spread along 0.25 and 0.75, with a concentration around 0.50. Therefore, the sensitivity parameter helps to relatively discriminate null and long-cycle behaviors.

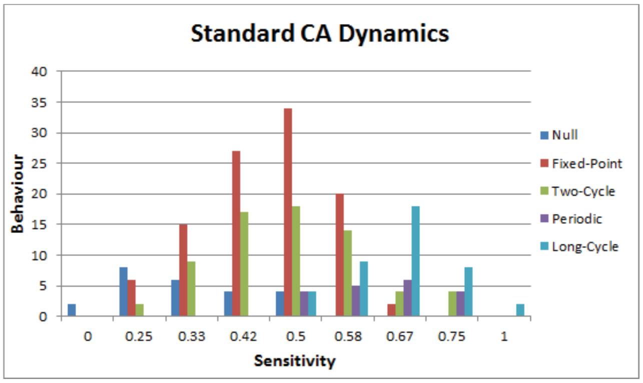


Fig.3. Sensitivity parameter and dynamical behaviors obtained for all the 256 rules in the elementary rule space.

Sensitivity was also used to perform a similar analysis using CA rules with pseudo-linear neighborhood. Each set is also formed by 256 rules, however, the structure of the neighborhood of each cell is defined according to the relations presented in the program graph used as reference, as showed in Figure 2 for the task 11. Gaussl8 program graph shown in Figure 1 was used to define the pseudo-linear neighborhood and the results indicate that the distinction obtained by sensitivity is not as perfect as in the linear neighborhood model as presented in Figure 4. However, it was possible to notice that the long-cycle behavior appears only when the value of sensitivity is greater than 0.5. Furthermore, most of the null rules have sensitivity less or equal to 0.5.

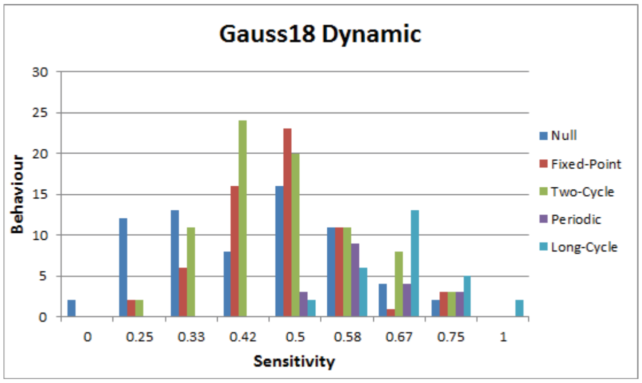


Fig.4. Sensitivity parameter and dynamic behaviors obtained for all the 256 rules with pseudo-linear neighborhood defined based on Gauss18.

A similar analysis was carried out in which the pseudo-linear neighborhood was elaborated based on a second program graph named Random30, which is presented in Figure 5. Using Random30 the distinction between null and long-cycle rules is higher than the observed using Gaussl8 as shown in Figure 6. It was also noticed a decay in the number of null rules (compared with Fig. 4).

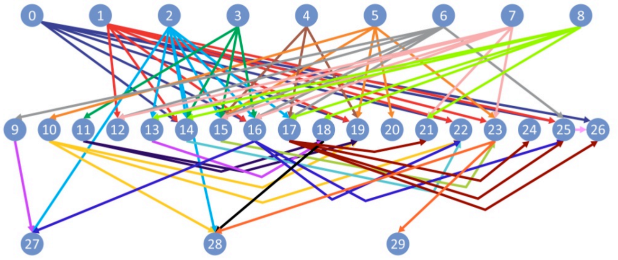


Fig.5**:** Program Graph Random 30.

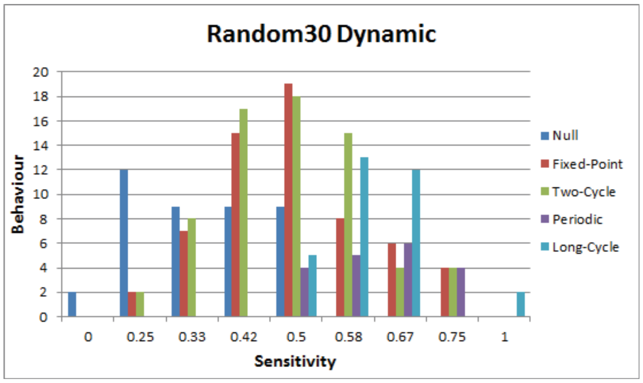


Fig.6. Sensitivity parameter and dynamic behaviors obtained for all the 256 rules with pseudo-linear neighborhood based on Random30.

Using the Sensitivity Parameter in the Search for Radius 2 CA-based Rules

Since it was possible to verify that sensitivity parameter can help to characterize radius 1 CA rules using both in linear as in non-linear neighborhoods, our next step was to investigate if the parameter could be used to help the genetic search for rules in a CA-based scheduler with an adequate dynamical behaviour. The new approach investigated here employs dynamical behavior forecast parameters to avoid CA rules with undesirable dynamics for scheduling propose. Similar approaches have been used in different CA applications (Oliveira et al. 2000). However, as far as we known, it has not been applied to CA-based scheduling. Particularly, the usage of dynamical forecast parameters had shown very effective for the density classification task (DCT) (de Oliveira et al. 2006). In those previous works, the information about the parameters was incorporated in the fitness evaluation of each individual of the population based on a desirable band of parameter values. However, in all previous works, the CA models investigated uses standard regular neighborhood. In the present work, we are interested to investigate if a parameter-based heuristic could also help the genetic search for adequate rules (in respect to the desirable dynamical behavior for the scheduling application) even if a non-regular neighborhood is employed in the CA model.

Analyzing the sensitivity of CA rules found by SCAS-HP

Once the analysis with radius 1 CA rules has shown that sensitivity parameter could help us to discriminate long-cycle and null behaviors as presented in last section, we started to analyze which kind of dynamical behavior could be associated to the rules find by SCAS-HP model (Carneiro and Oliveira, 2013) considering different program graphs to be scheduled. Therefore, we first replicated the SCAS-HP as described in (Carneiro and Oliveira, 2013) to see what kind of rules (in respect to their dynamics) was being returned by the genetic search during the learning phase. Table 1, 2 and 3 show the results obtained when the SCAS-HP was executed by 100 times for each program graph analyzed. Just the best rule of each run was considered. Three instances were considered for each program graph varying the number of processors (*P*) in the multiprocessor architecture (2, 3 and 4 processors) resulting in a different number of states per cell in the CA model (2, 3 and 4 states, respectively). Each table presents the results for each architecture: 2 processors (Table 1), 3 processors (Table 2) and 4 processors (Table 3). The program graphs used in such experiments were the same investigated in (Carneiro and Oliveira, 2013): gauss18 (Gs18), random30 (Rd30), random40 (Rd40) and random50 (Rd50), which have 18, 30, 40 and 50 tasks, respectively. Each table also shows the average value of sensitivity (μ) found calculating the parameter for the 100 best rules for each program graph. The tables show the number of rules (out of 100) that were classified in each dynamical class: Null (N), Fixed-Point (F), Two-Cycle (T), Short-Cycle (S) and Long-Cycle (L).

By analyzing the first line for each program graph (not in bold), it is important to notice that the dynamics profile of the evolved CA rules using SCAS-HP is very dependable on the program graph being scheduled. For example, in Table 1, it is possible to see that while less than 50% of the rules evolved to schedule Gauss18 are Long-Cycle, for the other three program graphs this kind of rule represents almost 100% of the observable behavior. The explanation of such behavior is that random graphs have much more non-linearity involved in their relations than the Gauss18 graph and this inherit non-linearity is expressed in their respective pseudo-linear neighborhoods, increasing the possibility to find CA rules with chaotic behavior (classified here inside the long-cycle class). In fact, when we performed an individual analysis of the rules evolved by SCAS-HP and automatically classified as long-cycle it was possible to observe that the vast majority of them can be classified as chaotic. Besides, it was possible to observe that the multiprocessor architecture analyzed (the number of processors) is also a relevant parameter to characterize the expectable dynamical behavior. When using 3 processors (or 3 states per cell), it is possible to observe in Table 2 that the number of long-cycle rules has a significant decay when compared with the 2-processors scenario (Table 1), especially for the random program graphs. However, when using 4 processors (Table 3), the number of long-cycle rules increases to above 90% of the rules evolved by SCAS-HP, except for the random50 graph.

Table 1: Dynamic behaviour of rules evolved by SCAS-HP and SCAS-HPμ for 2 processors architecture

Graph	Scheduler	μ_{AVG}	Dynamic Behaviour Class				
			N	F	T	S	L
Gs18	SCAS-HP	0,50	2	14	18	20	46
	SCAS-HPμ	**0,39**	**7**	**36**	**15**	**17**	**25**
Rd30	SCAS-HP	0,49	1	0	0	0	99
	SCAS-HPμ	**0,37**	**2**	**16**	**18**	**11**	**53**
Rd40	SCAS-HP	0,50	0	0	0	0	100
	SCAS-HPμ	**0,36**	**1**	**14**	**18**	**11**	**56**
Rd50	SCAS-HP	0,50	0	2	7	2	89
	SCAS-HPμ	**0,37**	**0**	**11**	**50**	**12**	**27**

On the other hand, when the sensitivity parameter of the rules is considered, it seems to be just dependable on the number of processors used, which reflects on the number of possible states per cell. For example, for 2 processors the average sensitivity is around 0.50, while it is around 0.64 with $P = 3$ and it is around 0.73 when 4 processors are used. Besides, the average sensitivity found in the experiments with SCAS-HP is almost the same for all the program graphs when the number of processors is fixed in a specific value. This observation is not trivial as the program graphs have different levels of non-linearity, as pointed before. It seems that SCAS-HP tends to find rules in a specific region of the parameter space, independently of the program graph employed to define the neighborhood relations.

Aiming to clarify this point we just generate a set of random rules to each rule space (P = 2, 3 and 4) to calculate the sensitivity parameter. We found that the sensitivity parameter associated to the random rules fits in specific intervals that we called the "natural bands" for each rule space (in respect to the number of states used). We noticed that more than 95% of the random rules fits in the following intervals: [0.45-0.55], [0.55-0.70] and [0.65-0.85], for P = 2, 3 and 4, respectively. We concluded that this increase of the sensitivity value with the increment of the number of processors/states is totally compatible with the definition of the parameter (Binder, 1994). As the number of states increases, as increases the possibility of a neighborhood configuration to be sensitive.

Besides, the sensitive values found for rules evolved by SCAS-HP also fit inside the natural bands calculated for the random samples. SCAS-HP evolution find rules in the same region of the natural bands: above 98% of the evolved rules is inside the natural bands (for each specification of number of states). What we concluded about this observation is that the GA starts from a random population of CA rules that tends to have sensitivity values inside the natural bands and after the evolution, even submitted to the pressure of the fitness evaluation based on their ability to schedule the program graph, the rules in the population still present sensitivity values inside the natural bands. Since we want to avoid the long-cycle rules and we have observed that this dynamical behavior is unlikely to observe in rules with sensitivity below the average value even using non-regular neighborhood (Carneiro and Oliveira, 2013), we decided to force the genetic search in SCAS-HP to search rules with lower sensitivity (below the natural bands).

Table 2: Dynamic behaviour of rules evolved by SCAS-HP and SCAS-HPμ for 3 processors architecture

Graph	Scheduler	μ_{AVG}	Dynamic Behaviour Class				
			N	F	T	S	L
Gs18	SCAS-HP	0,66	3	26	18	20	33
	SCAS-HPμ	**0,47**	**18**	**54**	**21**	**4**	**3**
Rd30	SCAS-HP	0,66	0	9	29	19	43
	SCAS-HPμ	**0,46**	**0**	**35**	**38**	**16**	**11**
Rd40	SCAS-HP	0,65	5	7	19	15	54
	SCAS-HPμ	**0,46**	**7**	**33**	**45**	**7**	**8**
Rd50	SCAS-HP	0,62	0	13	40	13	34
	SCAS-HPμ	**0,45**	**0**	**53**	**25**	**12**	**8**

Table 3: Dynamic behaviour of rules evolved by SCAS-HP and SCAS-HPμ for 4 processors architecture

Graph	Scheduler	μ_{AVG}	Dynamic Behaviour Class				
			N	F	T	S	L
Gs18	SCAS-HP	0,74	3	22	20	8	47
	SCAS-HPμ	**0,47**	**7**	**47**	**29**	**10**	**7**
Rd30	SCAS-HP	0,74	3	1	3	3	90
	SCAS-HPμ	**0,53**	**4**	**9**	**35**	**20**	**32**
Rd40	SCAS-HP	0,75	0	1	0	4	95
	SCAS-HPμ	**0,53**	**5**	**9**	**31**	**15**	**10**
Rd50	SCAS-HP	0,72	0	10	35	7	48
	SCAS-HPμ	**0,54**	**1**	**29**	**48**	**7**	**15**

SCAS-HPμ: a new CA-based scheduler guided by desired sensitivity bands

A new CA-based scheduling model was implemented based on the previous SCAS-HP model (Carneiro and Oliveira, 2013) incorporating a dynamical forecast parameter-based heuristic. This model also operates in learning and operation phases. The parameter-based heuristic was defined in a similar way for what was proposed in (Oliveira et al., 2000) to search CA rules for DCT. The original fitness function employed in SCAS-HP (Carneiro and Oliveira, 2013) was named P_M and it is given by the makespan associated to the allocation expressed by the final lattice found when the CA rule is applied over the lattice by 100 times starting from to the initial configuration. The parameter-based heuristic was coded as a function P_{μ}, which returns a value between 0 and P_M for each transition rule, depending on the values of the rule sensitivity in relation to a desired parameter band (S_1,S_2). Given a desired band (S_1,S_2) for the parameter, the value of P_{μ} for a specific rule r with sensitivity equal to μ_r is given by:

$$P_\mu(r) = \begin{cases} 0 & \text{if } S_1 \leq \mu_r \leq S_2 \\ min(|\mu_r - S_1|, |\mu_r - S_2|) \times P_M(r) & \text{otherwise} \end{cases}$$

It means that if the sensitivity rule is inside the desirable band the value of P_μ is zero. Otherwise, the value of P_μ in proportional to the makespan P_M and it is bigger as the sensitivity parameter is more distant of the desirable band. The genetic algorithm used in the learning phase in SCAS-HPμ was modified so as to incorporate the parameter-based heuristic in two aspects:

- The Fitness Function (*Fit*) of a cellular automaton rule was made by the weighted average of the original fitness function P_M and the parameter-based heuristic P_μ and it is given by:

$$Fit = P_M + \rho \times P_M$$

- Biased Reproduction and Mutation: in order to select the crossover point and the rule bits to be mutated, 10 attempts were made; among them, those that generated rules with high P_μ value were selected.

In the experiments reported here we used ρ = 0.4 and the following desirable bands for each architecture (number of processors *P*): 0.30 to 0.40 (*P* = 2); 0.40 to 0.50 (*P* = 3); and 0.45 to 0.55 (*P* = 4). In that way, the evolved rules are forced to be below the natural bands for each architecture. The second line (in bold) of each scenario named as SCAS-HPμ in tables 1, 2 and 3 presents the results when the new environment was used to evolve scheduler rules.

It is possible to observe that in all reported experiments with SCAS-HPμ the evolved rules returned an average sensitivity below the average value obtained without the parameter-based heuristic (SCAS-HP). As a consequence, the dynamics profiles had an expressive change using the new environment. Considering *P* = 2 experiments which returned the higher number of undesirable behaviour rules in the experiments with SCAS-HP, the number of long-cycle rules has decreased to the half (approximately). For *P* = 3 and 4 scenarios, the number of long-cycle rules is below 15% of the evolved rules (except for random30 with *P* = 4). Therefore, we can conclude that the use of parameter-based heuristic achieved the desired effect in the dynamical behavior of the evolved rules.

In a subsequent investigation, we analyzed if the pressure to be a low sensitivity rule has interfered in the ability of the evolved rules in the scheduling task. Table 4 shows the makespan of the best rule evolved for each scenario using SCAS-HP and SCAS- HPμ. One can observe that in the majority of the investigated scenarios, the best makespan did not change. In the scenarios in which the makespan have increased, this increment was below 5%.

The final investigation was performed to verify how the changes in the observed dynamical behavior could affect the performance of the evolved rules in the operation phase of the CA-based schedule. In such phase, it is expected that a CA rule evolved using a specific program graph in the learning phase could also have a good performance when this rule is applied to evolve other program graphs (not seen during the evolution). Table 5 shows the results obtained when the rules evolved using Random30 as the reference graph are applied to the other 3 graphs: Gauss18, Random 40 and Random 50. It is important to emphasize that in this phase there is no genetic evolution (or search); the pre-evolved CA rules are just applied for the new program graphs starting from the initial configuration defined by the selected heuristic. In our experiments DHLFET was used to initialize the lattices. Table 5 shows the best results found using the previous SCAS-HP scheduler model and the new one proposed here SCAS-HPμ, which uses the sensitivity parameter to avoid long-cycle rules. It is possible to observe that in almost all scenarios the rules evolved for Random30 graph by SCAS-HPμ returned a smaller makespam than the rules evolved for the same program graph by SCAS-HP. The two exceptions happened with the Random40 as the target graph using the architectures with 2 and 4 processors.

Table 4: Best makespan found by SCAS-HP and SCAS-HPμ (Learning Phase)

	SCAS-HP			**SCAS-HPμ**		
Graph	**2**	**3**	**4**	**2**	**3**	**4**
Gs18	44	44	44	44	44	44
Rd30	1222	847	788	1222	**865**	788
Rd40	983	681	564	983	**688**	564
Rd50	624	528	500	624	**548**	**508**

Table 5: Best makespam obtained after the application of CA rules found for Random30 program graph by SCAS-HP and SCAS-HPμ to new program graphs (Operation Phase)

Graph	**P**	**SCAS-HP**	**SCAS-HPμ**
Gs18	2	49	**47**
	3	50	**48**
	4	47	**44**
Rd40	2	**996**	997
	3	762	**750**
	4	**655**	658
Rd50	2	684	**664**
	3	656	**576**
	4	684	**600**

Conclusion and future work

Starting from a previous CA-based scheduler named SCAS-HP (Carneiro and Oliveira, 2013) we proposed a new model named SCAS-HPμ. In this new model, the evolutionary search used in the learning phase, in which cellular automata rules are evolved to schedule a specific program graph, is guided by a dynamics forecast parameter to find not only the distribution of tasks associated to the smallest makespam to the graph, but also an adequate dynamical behavior. The major motivation to try to control the dynamical behavior of the evolved rules is because it was possible to observe in our experiments that the non-regular neighborhood employed in SCAS-HP (Carneiro and Oliveira, 2013) push the evolutionary search to find chaotic rules in a considerable number of executions, specially when the program graph to be scheduled has inherit non-linearity. Besides, we believe that chaotic rules do not do an effective scheduling of the program graph and they are select by the evolutionary search just by the causality to generate a random lattice which represents a schedule with a reasonable makespam to the problem. Probably, these chaotic rules will not return a good schedule when the evolved rules will be applied to new program graphs during the operation phase of the model.

Our results shown that the employment of the forecast parameter named sensitivity (μ) proposed in (Binder, 1988) for standard CA with regular neighborhood is also applicable for non-regular neighborhood to characterize long-cycle rules (a superclass into which the chaotic class belongs). It was possible to elaborate a heuristic based on the desirable bands of μ to avoid the long-cycle behavior. The learning phase of the new model SCAS-HPμ uses this parameter-based heuristic to guide a genetic algorithm to find rules that exhibits the desirable dynamical behavior (not long-cycle) at the same time that are able to find a good scheduling for a given program graph. Moreover, when these evolved rules were applied to new program graphs in the operation phase of the model, they were able to find better schedules for a set of new program graphs, when compared to the previous model SCAS-HP (Carneiro and Oliveira, 2013). The analysis of the evolved rules in the operation phase is important because their performance anticipate the expected behavior of the CA-based model when it is applied to real instances of processor task scheduling. That is, when a new instance of a program graph is presented as an input of the CA-based scheduler, it will use pre-evolved transition rules from a repository to solve it.

Although the results presented here are very promising, we still need to deeper analyze the relationship between the dynamical behavior of the evolved rules and their performance both in learning and operation phases. As presented in Table 5, in two out of nine scenarios investigated, the results obtained with the rules evolved with SCAS-HP (which were predominantly chaotic) returned better results than the rules evolved with the new model (which provoked a severe decay in the number of chaotic rules). Similar experiments performed with other program graphs showed that this situation can happen in other scenarios. Therefore, it is important to perform additional experiments to understand how a group dominated by chaotic behavior can lead to better results in some scenarios. However, we can affirm that in general, the results obtained with the new model overcame the previous model in the operation phase, as we expected after controlling the dynamical behavior.

As future research we intend to investigate different parameter bands to elaborate the heuristic to guide the evolutionary search. Some preliminary experiments show that the better sensitivity desirable band can vary from one program graph to another. We also intend to investigate the usage of other parameters in the scheduling model as the neighborhood dominance proposed in (Oliveira et al. 2001).

Besides, two limitations of the CA-based scheduling approach discussed here are: (i) the need to predefine the desirable parameter band; (ii) the difficult to deal with architectures using a high-number of processors. As a future investigation we intend to address both limitations.

AKNOWLEDGMENTS

TI nad GMBO are grateful to Fapemig, CNPq and CAPES.

References

Adam, T. L., Chandy, K. M., and Dickson, J. R. (1974). A comparison of list schedules for parallel processing systems. Commun. ACM 17, 685-690.

Binder, P.M. (1994). Parametric Ordering of Complex Systems. Physics Rev. E.

Carneiro, M. and Oliveira, G (2011). Cellular automata-based model with synchronous updating for task static scheduling. In Proceedings of 17th International Workshop on Cellular Automata and Discrete Complex System (Automata 2011), Chile.

Carneiro, M. and Oliveira, G. (2013). Synchronous cellular automata-based scheduler initialized by heuristic and modeled by a pseudo-linear neighborhood. Natural Computing, 12(3), 339-351.

Cosnard, M., Marrakchi, M., Robert, Y., e Trystram, D. (1988). Parallel Gaussian elimination on an MIMD computer. Parallel Computing, 6(3):275-296.

De Jong, K. Evolutionary computation: a unified approach. MIT Press, 2006.

De Oliveira, P., Bortot, J. and Oliveira, G. (2006). The best currently known class of dynamically equivalent cellular automata rules for density classification, Neurocomputing, Vol. 70, Pages 35-43.

Li, Wentian and Packard, N. (1990). The Structure of Elementary Cellular Automata Rule Space. Complex Systems, 4:281-297.

Mitchell, M. Computation in cellular automata: A selected review. Non-standard Computation, pages 385-390, 1996.

Oliveira, G, de Oliveira, P., Omar, N. (2000). Evolving solutions of the density classification task in 1D cellular automata, guided by parameters that estimate their dynamic behavior. In: Artificial Life VII. pp. 428-436.

Oliveira, G., de Oliveira, P. and Omar, N.. (2001) Definition and application of a five-parameter characterization of one-dimensional cellular automata rule space. Artificial Life 7(3): 277-301.

Oliveira, G. and Vidica. P. (2012). A Coevolutionary Approach to Cellular Automata-Based Task Scheduling. Proc. of ACRI2012, LNCS 7495, pp 111-120.

Seredynski, F., and Zomaya, (2002). A. Sequential and parallel cellular automata-based scheduling algorithms. Parallel and Distributed Systems, IEEE Transactions on 13(10): 1009-1023.

Swiecicka, A., Seredynski, F. and Zomaya, A. . (2006). Multiprocessor scheduling and rescheduling with use of cellular automata and artificial immune system support. Parallel and Distributed Systems, IEEE Transactions on, 17(3), 253-262.

Wolfram, S. (1994). Cellular Automata and Complexity. U.S.A.: Addison-Wesley**.**

Investigating the Impact of Communication Quality on Evolving Populations of Artificial Life Agents

Sadat Chowdhury[1] and Elizabeth I. Sklar[2]

[1]The Graduate Center, The City University of New York, New York, NY, USA
[2]Department of Computer Science, University of Liverpool, Liverpool, UK
sadatc@gmail.com,e.i.sklar@liverpool.ac.uk

Abstract

The work presented in this paper examines the relationship between *interaction mechanism* and *heterogeneity* in evolutionary multiagent systems. Three interaction mechanisms are explored (stigmergy, broadcast and unicast), in both homogenous and heterogenous populations of artificial agents. A number of different schemes are compared in an experimental environment which concerns agents that evolve to detect, extract and collect resources in a grid-based world. A series of experiments was conducted to analyze the effects of interaction mechanism and population diversity on multiple performance metrics, under various signal transmission qualities. The results show differences in key performance metrics when signal transmission quality degrades, under some experimental conditions. These observations offer an important take-away message for assessing the robustness of systems that may face inconsistent communication network quality, a common problem in today's "wired" real world.

Introduction

In our research, we are investigating the relationship between *interaction mechanism* and *heterogeneity* in evolving multiagent populations. Our work extends earlier experiments conducted by Sklar et al. (2006) in which it was hypothesized that the performance of a multiagent team is affected by both population diversity and level of interaction, such that if interaction mechanisms were employed effectively, a population of heterogenous, single-function agents could generally outperform a homogeneous population of multi-function agents. Several sets of experiments were conducted; and although trends appeared to confirm the hypothesis, the results were statistically inconclusive due to large variances in the experimental data. Here we continue this line of work, in a new version of the simulation environment, *synthScape* (Chowdhury and Sklar, 2015), which runs on a *high-performance computing cluster (HPCC)* so that we can obtain statistically significant results.

The focus of this paper is on evaluating the robustness of interaction mechanisms and population heterogeneity in the face of fluctuation in communication signal. Three interaction mechanisms are explored: *stigmergy (trail)*, *broadcast* and *unicast*. These are tested in two types of evolving populations of artificial agents: *homogenous* and *heterogenous*. The agents collect *resources* from geospatially dispersed locations and bring them to designated *collection sites*. This involves a sequence of *detection*, *extraction* and *transportation* tasks on the part of the agents. The same agent need not be responsible for completing all three tasks; indeed, having the tasks completed simultaneously by different coordinating agents may produce more efficient solutions. Coordination may be achieved by agents *communicating* to indicate that one task in the sequence has been completed at a particular location and the next task can be attempted.

Although the problem space we consider here involves three sequential tasks, our environment and framework are defined to generalise to n-sequential-task domains. The complexity of solutions in sequential-task domains increases with the number of tasks. For example, a foraging problem that requires additional tasks such as *refinement* followed by *processing* of resources, between extraction and transportation, is a more complex 5-task problem than our baseline 3-task version. Each type of task requires a corresponding *trait*, or agent capability, and, in general, solving k-task problems in the sequential-task domain will require k traits. Such problems can be solved by heterogenous populations of simpler specialist agents or homogenous populations of more complex generalist agents. Stated formally, a k-task problem in the sequential task domain can be solved either by: (a) a heterogenous population of n-trait agents where $n \leq k$; or (b) a homogenous population of k-trait agents[1].

The problem of *communication network quality* is real and significant in today's "wifi" society. In the artificial life, multiagent systems and robotics communities, many solutions to various problems are devised and tested in simulation before being deployed an the "real" world. One common problem with transferring solutions to real-world settings is that certain features which work perfectly in simulation are problematic in the real world and can cause the breakdown of an approach that performs well in simula-

[1] When $n > k$, agents have additional capabilities that are not used in solving problems in the domain.

tion. One such feature is *communication*. We have all experienced communication failure when our laptop or mobile phone loses its signal and our connection dies. The work presented here examines the impact of communication quality on several different versions of our evolving multiagent system.

The remainder of this paper is organised as follows. First, we review some related work, specifically research in the literature that has defined taxonomies for interaction or communication mechanisms in multiagent environments. Second, we describe how our experimental environment, *synthScape*, operates. Then we detail our experiments and present results. Finally, we close with discussion.

Related Work

Interactions are an essential requirement for most complex coordination tasks in any *multiagent system (MAS)* (Wooldridge, 2002). Interactions, however, require varying degrees of effort by the agents or specific environmental conditions or both.

Coordination among agents can be achieved using non-interactive techniques, such as tacit agreements, conventions, and simple rules, as has been shown in the famous boid examples in Reynolds (1987). Studies with evolutionary agents, such as Haynes et al. (1995b) and Haynes et al. (1995a), show that simple coordination can be achieved with co-evolved agents without any explicit interactions. The researchers of these studies concluded, however, that such non-interactive techniques are either insufficient or inefficient for multiagent coordination tasks in general. In Campbell et al. (2008), it is shown that non-interactive techniques work well in situations where the ratio of task length to agent team size is small, but their performance decreases as this ratio increases.

More complex coordination among agents can be achieved using indirect forms of interaction mechanisms such as stigmergy. It has been observed that social insects, such as bees and ants, use chemical trails, called *pheromones* that modify the environment and can later be detected by other members of their own species. The modifications (chemical trails) are used as a form of communication; this process of *stigmergy* was first described in Grasse (1959). The concept of stigmergy was used as a basis for agent coordination in techniques such as *ant colony optimization (ACO)* (Dorigo et al., 2000) and *swarm intelligence (SI)* (Dorigo et al., 2006). A real-world use of stigmergy in an industrial application is described in Valckenaers et al. (2004). A physical implementation of stigmergy on real robots was reported in Sahraei et al. (2013).

An even wider range of complex coordination problems can be solved more efficiently with direct interaction mechanisms such as explicit communication. This has been demonstrated in numerous studies (Barlow et al., 2008; Naeini and Ghaziasgar, 2009; Doherty and O'Riordan, 2009). In general, these studies compare the efficiency of overall coordination with and without communication capabilities between the agents and demonstrate the positive aspects of explicit communication.

There have been several attempts to classify interaction mechanisms. In Deugo et al. (2001), several coordination (interaction) mechanisms are presented in the form of software pattern guidelines for developers of MAS. In Menge (1995), and more recently in Eguchi et al. (2006), characteristics that dictate agent interaction behavior is presented; these characteristics can be used as a way to classify interaction mechanisms. Eguchi et al. (2006) lists these characteristics in the form of interaction "attitudes" (shown in Table 1) that an agent (Agent *s*) takes towards another agent (Agent *o*). For example, an agent having a *mutualism* attitude towards another agent will interact with that agent in a way that will benefit both agents.

Attitude	*Agent s*	*Agent o*
Mutualism	Improve	Improve
Harm	Deteriorate	Deteriorate
Predation	Improve	Deteriorate
Altruism	Deteriorate	Improve
Self Improvement	Improve	—
Self Deterioration	Deteriorate	—

Table 1: Interaction Attitudes from Eguchi et al. (2006)

Sklar et al. (2006) lists the following taxonomy classes of interaction mechanisms that form the basis of the interaction mechanisms used in this work:

Tacit agreements: There is no explicit communication between agents, and instead social norms or pre-determined rules govern agent behavior. An example of such an interaction is the flight patterns of flocks of migratory birds: each bird tries to maintain an average distance from other birds and tries to fly in the same general direction as the rest of the flock.

Environmental cues: Agents modify the environment in such a way that another agent can detect and act on that modification. An example of such an interaction is that of an ant leaving a trail of pheromones that others can follow.

Signal broadcasting: Agents explicitly broadcasts signals for other agents to receive. This is basically direct inter-agent communication. Table 2 shows parameter values for these interaction mechanisms.

Additional classifications and taxonomies can be found in Wooldridge (2002) and Weiss (1999). There have been studies where classifications were made based on the role played by the environment (Keil and Goldin, 2006) and the information available in the system (Parunak et al.,

Parameter	Tacit Agreement	Environmental Cues	Signal Broadcasting
Interaction Medium	—	environmental	broadcast
Message Content	—	scalar	binary/ scalar
Message Frequency	—	continual	continual

Table 2: Taxonomy classes of interaction mechanisms in Sklar et al. (2006)

2004). In addition, the evolution of agent signaling from interactions—especially signals with semantic content and language-like properties—has been the subject of a number of studies (Tuci, 2009; Nehaniv, 2000, 2005; Skyrms, 2010).

Durfee (2001, 2004) lists stress factors that need to be addressed when scaling up multiagent systems along various dimensions in order to solve increasingly complex problems (Table 3). *Heterogeneity*, *degree of interaction* and *efficiency*—all dimensions that we consider in our work presented here—are seen as important along the stress factors listed.

Factor	*Dimensions*
Agent Population	quantity, complexity, heterogeneity
Task Environment	degree of interaction, distributivity, dynamics
Solution	efficiency, quality, robustness, overhead limitations

Table 3: Multiagent Coordination Stress Factors from Durfee (2001)

Curran and O'Riordan (2006) examine the effects of interactions in the form of cultural learning on both fitness and diversity. Their results indicate that the addition of such interactions (cultural learning) promotes fitness and significantly increases both genotypic and phenotypic diversity in the population. More recent work on language and communication evolution in embodied agent can be found in (Nolfi and Mirolli, 2009; Szathmáry, 2010).

Here we present a methodical study which investigates the relationship between type of interaction mechanism and population heterogeneity, with respect to communication quality.

Experimental Environment

As mentioned above, our experimental environment is called *synthScape*. This framework was developed in *MASON*, a discrete event simulation platform (Luke et al., 2005). Populations of agents are evolved using *genetic programming (GP)* techniques. The control program (*genotype*) of an individual agent is constructed from a custom extension of the Push language (Spector, 2001; Spector et al., 2005). Push programs are series of instructions that are executed by a stack-based virtual machine. Push was designed specifically for evolutionary computation, and thus has two big advantages: (1) any set of Push instructions makes a valid program, so that genetic reproduction operators cannot produce incomplete or invalid programs; and (2) programs can modify themselves by allowing instructions to manipulate the code stack (where instructions reside), which can potentially introduce complex control strategies. In our implementation, each instruction is atomic and either pushes values onto a code stack, executes instruction(s) from the code stack, or executes domain-specific instructions (i.e., sense, move, communicate). Because of the intense computational requirements necessary to produce statistically significant results, *synthScape* has been designed to be distributed across several processors in a HPCC[2], where each node in the HPCC executes one experimental scenario at a time.

The experiments described in the next section explore communication quality for six different types of populations. Each experiment used the following parameters:

dimensions of simulated world	=		16×16
population size	=		$\{8, 24\}$
resource capture goal	=	12	(75%)
obstacle density	=	32	(12.5%)
resource density	=	16	(6.25%)
collection site density	=	4	(1.56%)

Each population is characterised by: *interaction mechanism* and *heterogeneity*. Populations evolve using an evolutionary algorithm inspired by (Watson et al., 1999; Bianco and Nolfi, 2004), where the concept of evolution is embodied within the agents. An agent maintains its own gene pool, runs its own evolutionary algorithm and evolves its own genotype with a particular set of traits. When the control code (genotype) has finished its execution, it is replaced by the next generation's control code. Agents belonging to the same species, situated in close proximity, are able to copy each others' genotypes to add to their own gene pools. The concept is similar to mating in nature and may allow the transfer of useful genotypes across gene pools and unify the behaviour of agents belonging to the same species. Other scale-related benefits can be realised through an extended form of this basic model (Laredo et al., 2011).

Different types of **interaction mechanisms** facilitate communication between the evolving agents. During an interaction, signals are transferred from agents in "sender" mode to agents in "receiver" mode. The receiving agents

[2] `http://wiki.csi.cuny.edu/cunyhpc/`

may ignore the signals or decide to act upon them. Here, we employ signals with elementary semantics that indicate the state of detected resources and their location. The resources can be in one of three states: FOUND-STATE (thus, ready to be extracted), EXTRACTED-STATE (thus, ready to be processed), or PROCESSED-STATE (thus, ready to be transported). Three interaction mechanisms are considered here: ***trail***—where senders have the ability to leave "trail signals" on the grid, a form of stigmergy; ***broadcast***—where senders have the ability to broadcast signals to receivers within a certain range, who receive the signals instantly; and ***unicast***—where senders have the ability to send signals to the closest receiver.

Different levels of **heterogeneity** within a population represent the ability of individual agents to perform one or more tasks. We define the heterogeneity of a population based on three factors: *species richness*, *species evenness* and *species diversity*. Species richness, S, is the number of unique species in a population. Species evenness (Peet, 1974; Mulder et al., 2004), E, is based on Simpson's dominance index:

$$E = H/ln(S)$$

where H, the measure of species diversity, is based on Shannon's entropy (Shannon, 1948):

$$H = -\sum_{i=1}^{S} p_i ln(p_i)$$

and p_i is the proportion of species i. E can range from 0, when there is only one dominant species in the population, to 1, when all species are equally abundant in the population. Higher values of H are indicative of both greater richness and evenness in the population; if there is only one species, H approaches 0. The two types of populations considered in the experiments we present here are: ***heterogenous***, where $(S, E, H) = (3, 1, 1.098)$; and ***homogenous***, where $(S, E, H) = (1, 0, 0)$.

Experiments

The experiments conducted here examine the impact of communication quality on six different evolutionary multi-agent populations. The different populations are pairwise combinations of the 3 interaction mechanisms (trail, broadcast and unicast) and 2 population heterogeneity levels (homogeneous and heterogeneous), described in the previous section.

Below are some additional details explaining how each interaction mechanism works:

- *trail:* an agent can drop a trail signal in a cell that any other agent moving through the same cell can detect. Trail signal strength deteriorates over time (currently set to dissipate at the rate of 60% at every step) and does not convey additional semantics to the receiving agent. Receiving agents can: (a) acknowledge that a trail was detected and set an internal state variable or (b) move towards a neighboring cell containing the highest concentration of signal strength.
- *broadcast:* an agent can send one of two signals: SIGNAL-A and SIGNAL-B. Once a signal is transmitted, any agent can receive that signal within 2 time steps of the simulation. Receiving agents can: (a) acknowledge it and set an internal state variable or (b) move towards the source of the signal.
- *unicast:* an agent can send one of two signals: SIGNAL-A and SIGNAL-B to the agent that is closest to it. Receiving agents can: (a) acknowledge it and set an internal state variable or (b) move towards the source of the signal.

Note that the meaning of the generic SIGNAL-A and SIGNAL-B depends on which agent transmits the signal (see below).

A number of *constraints* were placed on agents' actions when the broadcast and unicast interaction were used. These are as follows:

- An agent with detection capability is only allowed to send SIGNAL-A.
- An agent with the extraction capability is only allowed to extract a resource once it has received SIGNAL-A. An extractor agent may also send SIGNAL-B to announce that there is an extracted resource that is ready to be collected.
- An agent with the transportation capability is only allowed to load, carry and unload an extracted resource—potentially to a collection site—once it has received SIGNAL-B.

These constraints ensure that the agents' actions are instigated as the result of receiving a communication signal, as opposed to occurring randomly (e.g., because an extractor agent randomly stumbled on a cell where a found resource was ready to be extracted).

Within each of the six interaction-mechanism/population-heterogeneity combinations evaluated, the communication signal quality was varied. Two extreme values were tested: 100% was the highest quality, meaning that no signals were lost; and 25% was the poorest quality, meaning that three-quarters of signals transmitted were lost. Two intermediate values, 75% and 50%, were also tested, indicating one-quarter and one-half of the signals transmitted were lost, respectively.

Results

The following section describes the results of our experiments. Each of the 6 populations tested was evolved for 1000 generations. Using the HPCC, 100 runs were executed for each. Average results are presented.

We examine the results in terms of 4 metrics: (a) resource capture rate; (b) interaction instruction transmission rate; (c) signals received; and (d) resource collection interval. Each is presented below.

Resource Capture Rate. Figure 1 contains the *resource capture rates* for all 6 experimental conditions. This metric indicates the percentage of resources in the agents' environment which were successfully detected, extracted and transported. We make several observations about our results. First, both homogenous and heterogenous populations evolve to near 100% capture rate when interacting with trail signals. Second, the heterogenous agents evolve towards optimal capture rate much faster than the homogenous agents. This is more apparent when the agents communicate using broadcast and unicast. Third, the behavior of both populations using the *trail* interaction mechanism are very similar. One subtle difference is that the heterogenous agents achieve 50% capture rate within 150 generations, whereas it takes close to 300 generations in the case of the homogenous agents. Both populations seem to be not very sensitive to the signal transmission quality. Fourth, in both populations, signal transmission quality in the *broadcast* interaction mechanism positively impacts the capture rate. Overall, the heterogenous agents perform better than the homogenous agents (seemingly by some constant factor). Fifth, the performance of both populations with the *unicast* interaction mechanism mirrors that of broadcast, but seems to be reduced by some constant factor in both populations.

Next, we examine the ranking of the results. The best performance came from heterogenous agents using trails, irrespective of the quality of the communication signal. The second-best performance came from homogenous agents using trails, again irrespective of the quality of signal. The third-best performance came from heterogenous agents using broadcast with 100% signal transmission quality. The fourth-place performance came from heterogenous agents using unicast with 100% signal transmission quality. The worst performance came from homogenous agents using broadcast and unicast with 25% signal transmission quality.

In conclusion, with respect to resource capture rate, the heterogenous agents with trail perform best overall. There might be situations where trail is not an ideal form of communication. For example, trails can be arbitrarily removed by the environment, and there can be "cross-talk" (where different signals conflict).

Interaction Instruction Transmission Rate. Figure 2 plots the number of interaction instructions—send and receive commands—that were transmitted by the agents as they were evolving. We make several observations about our results. First, the heterogenous agents transmit significantly higher numbers of interaction instructions. In the case of trail, homogenous agents evolve to practically not transmitting any send or receive trails—even though they can utilize the signals. Second, the heterogenous-trail agents settle on a fairly consistent transmission rate (e.g., 100 instructions at the signal transmission rate of 100%). Third, in both populations using the broadcast interaction mechanism, signal transmission quality positively impacts the rate of issuing interaction. Overall, the heterogenous agents issue significantly more instructions than homogenous agents. Fourth, with the heterogenous-unicast agents, the signal transmission quality seems to have little or no impact on the rate of transmissing interaction instructions. Although the signal transmission quality is somewhat differentiated by the homogenous population, it issues significantly fewer instructions. Fifth, the number of instructions transmitted seems to steadily rise over generations for both broadcast and unicast interaction mechanisms. Sixth, in the heterogenous populations, there is an initial spike, followed by a sharp decline, and then a steadier rise of interaction instructions. One possible explanation for this behavior is that in the initial generations, communication behavior provides a high fitness and spreads among the population; however, there is a sharp decline once the population becomes more selective in its use of communication instructions. In the homogenous population, where communication plays less role, this dipping behavior is absent.

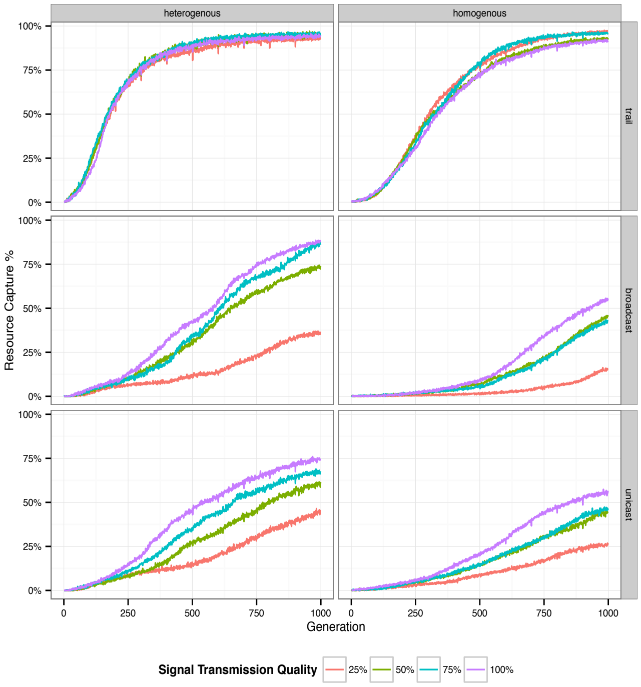


Figure 1: Resource Capture rate (y-axis).

Next, we examine the ranking of the results. The best results were obtained by heterogenous agents using broadcast with higher signal transmission rates. The second-best results were obtained by heterogenous agents using unicast, irrespective of the signal transmission rate.

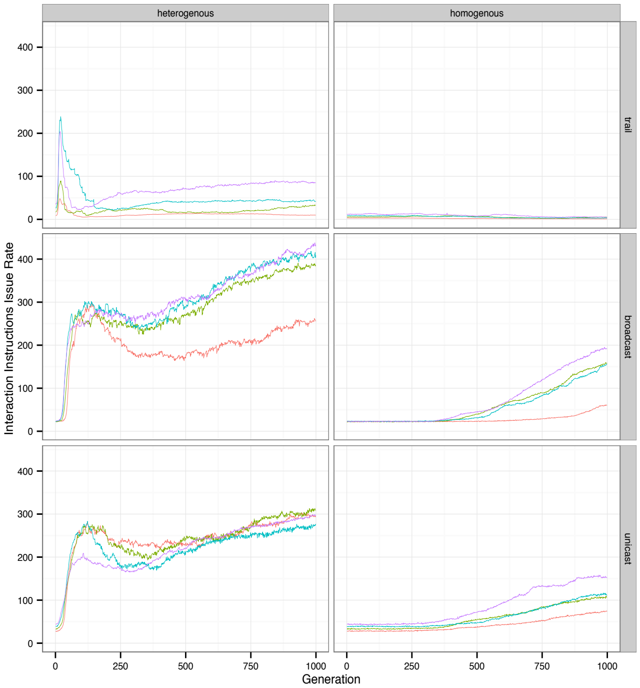


Figure 2: Transmission Rate (y-axis). Legend is the same as in Figure 1.

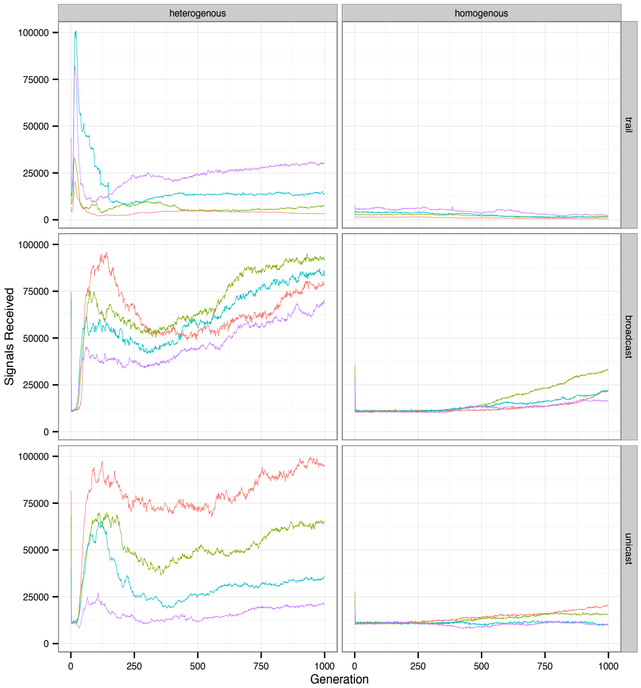


Figure 3: Signals Received (y-axis). Legend is the same as in Figure 1.

Signals Received. Figure 3 contains the number of signals that are actually received in each generation. We make several observations about our results. First, for all interaction mechanisms, heterogenous agents evolve to receive more signals than the homogeneous agents. Second, when heterogenous agents use the trail mechanism, they evolve to receive more signals when the signal transmission rate is higher; whereas, in the case of broadcast and unicast, the agents evolve to receive fewer signals. This suggests that with trail, the receipt of more signals results in better optimization, whereas with broadcast and unicast, receipt of fewer signals results in better optimization.

Resource Collection Interval. Figure 4 shows the evolution of the average number of steps between resource collections. We make several observations about our results. First, the number of steps settles down to a smooth curve very quickly in the case of heterogenous agents for all interaction mechanisms, as compared to homogeneous agents. Second, the plots show how much the transmission quality impacts the average collection interval in both homogenous and heterogenous agents. In all cases, a high transmission quality significantly lowers the collection interval and keeps the value low across the generation (there is less jitter). Third, the heterogenous-trail agents settle down to a consistently lower interval much more quickly, regardless of the transmission quality.

Next we examine the ranking of the results. The best results were obtained by the heterogenous and homogenous agents using the trail interaction mechanism. The second-best results were obtained by the heterogenous-unicast agents with 100% transmission quality. The third-best results were obtained by the heterogenous-broadcast agents with 100% transmission quality.

Discussion

A brief discussion of our overall results follows.

First, the best task completion rates (capture rate) are evolved by both the homogenous and heterogenous agents using the trail interaction mechanism. Additionally, heterogenous agents evolve to higher efficiency levels faster than homogenous agents.

Second, homogenous agents evolve to use much less communication than heterogeneous agents. Our conclusion is that communication adds extra overhead that homogenous agents tend to keep to a minimum, because they can perform all the tasks themselves (i.e., without needing the cooperation of other agents, like the heterogenous agents do).

Third, in this constrained version of our environment, heterogenous agents are forced to communicate in order to capture resources. The result is that they achieve higher efficiency levels than in our previous work, which was conducted in an unconstrained version of our environment (Chowdhury and Sklar, 2015).

Fourth, the signal transmission quality overall has a pos-

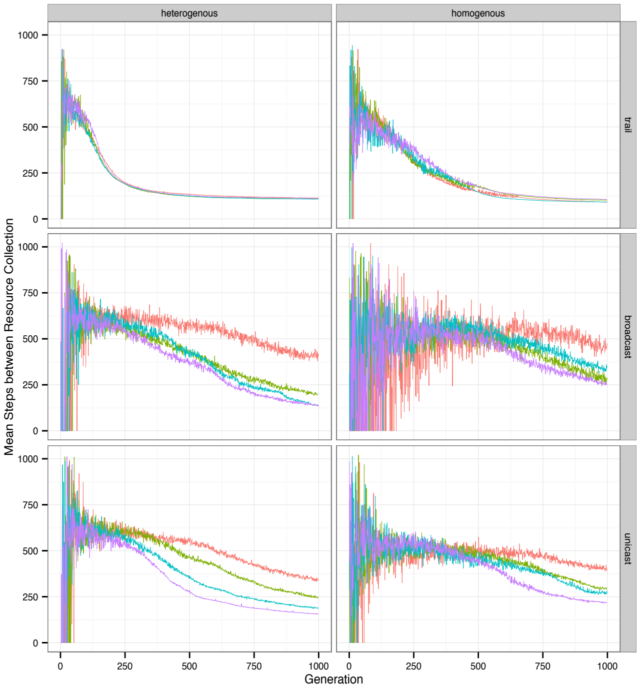


Figure 4: Interval Between Resource Collections. Legend is the same as in Figure 1.

itive impact on the task completion rate. In other words, the better the signal transmission quality, the higher the task completion rate.

Fifth, broadcast and unicast receivers evolve to spend less instructions to receive signals when the transmission rates are better. In the case of the trail mechanism, when transmission rates are better, more "receive signal" instructions are issued. This makes sense: trail signals do not convey any semantics—they are generic messages. In contrast, broadcast and unicast signals convey richer semantics: for example, whether something is an extractable resource.

Finally, aside from the trail mechanism, unicast with 100% signal transmission quality evolves the best (lowest) resource collection interval. This also makes sense: unicast is sent to the closest agent and this should theoretically lower the collection times.

In conclusion, we have shown that the communication quality has an impact on key performance metrics in an evolutionary multiagent environment applied to a 3-task sequential domain. However, the overall impact of communication is not necessarily positive and is dependent on the heterogeneity of the population. The impact is more significant with broadcast and unicast modes of interaction in the heterogenous populations; these interaction mechanisms are richer from a semantic standpoint with respect to message content. The trail mode of interaction is more resilient to degradation in quality of communication in both heterogenous and homogenous populations.

Our next steps with this line of work will examine how diversity impacts performance, for example how various species diversity, evenness and richness values effect performance.

Acknowledgements

This research was supported, in part, by a grant of computer time from the City University of New York High Performance Computing Center under NSF Grants CNS-0855217, CNS-0958379 and ACI-1126113; as well as a University of Liverpool Research Fellowship.

References

Barlow, G. J., Oh, C. K., and Smith, S. F. (2008). Evolving cooperative control on sparsely distributed tasks for UAV teams without global communication. In Keijzer, M., Antoniol, G., Congdon, C. B., Deb, K., Doerr, B., Hansen, N., Holmes, J. H., Hornby, G. S., Howard, D., Kennedy, J., Kumar, S., Lobo, F. G., Miller, J. F., Moore, J., Neumann, F., Pelikan, M., Pollack, J., Sastry, K., Stanley, K., Stoica, A., Talbi, E.-G., and Wegener, I., editors, *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 177–184, Atlanta, GA, USA. ACM.

Bianco, R. and Nolfi, S. (2004). Toward open-ended evolutionary robotics: evolving elementary robotic units able to self-assemble and self-reproduce. *Connect. Sci.*, 16(4):227–248.

Campbell, A., Wu, A. S., and Shumaker, R. (2008). Multi-agent task allocation: Learning when to say no. In *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, GECCO '08, pages 201–208, New York, NY, USA. ACM.

Chowdhury, S. and Sklar, E. I. (2015). Exploring interaction, diversity and efficiency of biologically inspired evolutionary multiagent systems. In *8th International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*. ICST.

Curran, D. and O'Riordan, C. (2006). Increasing population diversity through cultural learning. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 14(4):315–338.

Deugo, D., Weiss, M., Kendall, E., et al. (2001). Reusable patterns for agent coordination. *Coordination of Internet Agents, Springer*.

Doherty, D. and O'Riordan, C. (2009). Effects of shared perception on the evolution of squad behaviors. *IEEE Transactions on Computational Intelligence and AI in Games*, 1(1):50–62.

Dorigo, M., Birattari, M., and Stutzle, T. (2006). Ant colony optimization. *Computational Intelligence Magazine, IEEE*, 1(4):28–39.

Dorigo, M., Bonabeau, E., and Theraulaz, G. (2000). Ant algorithms and stigmergy. *Future Generation Computer Systems*, 16(8):851–871.

Durfee, E. (2001). Scaling up agent coordination strategies. *Computer*, 34(7):39–46.

Durfee, E. (2004). Challenges to scaling-up agent coordination strategies. *An Application Science for Multi-Agent Systems*, pages 113–132.

Eguchi, T., Hirasawa, K., Hu, J., and Ota, N. (2006). A study of evolutionary multiagent models based on symbiosis. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 36(1):179–193.

Grasse, P. P. (1959). La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. la theorie de la stigmergie: essai d'interpretation du comportement des termites constructeurs. *Insectes Sociaux*, 6:41–81.

Haynes, T., Sen, S., Schoenefeld, D., and Wainwright, R. (1995a). Evolving multiagent coordination strategies with genetic programming. Technical Report UTULSA-MCS-95-04, The University of Tulsa.

Haynes, T., Sen, S., Sen, I., Schoenefeld, D., and Wainwright, R. (1995b). Evolving a team. In *In Working Notes of the AAAI-95 Fall Symposium on Genetic Programming*, pages 23–30. AAAI.

Keil, D. and Goldin, D. (2006). Indirect interaction in environments for multi-agent systems. In *Proceedings of the 2nd international conference on Environments for Multi-Agent Systems*, E4MAS'05, pages 68–87, Berlin, Heidelberg. Springer-Verlag.

Laredo, J. L. J., Lombraña González, D., Fernández de Vega, F., Arenas, M. G., and Merelo Guervós, J. J. (2011). A peer-to-peer approach to genetic programming. In Silva, S., Foster, J. A., Nicolau, M., Giacobini, M., and Machado, P., editors, *Proceedings of the 14th European Conference on Genetic Programming, EuroGP 2011*, volume 6621 of *LNCS*, pages 109–118, Turin, Italy. Springer Verlag.

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., and Balan, G. (2005). Mason: A multiagent simulation environment. *Simulation*, 81(7):517–527.

Menge, B. (1995). Indirect effects in marine rocky intertidal interaction webs: patterns and importance. *Ecological monographs*, 65(1):21–74.

Mulder, C. P. H., Bazeley-White, E., Dimitrakopoulos, P. G., Hector, A., Scherer-Lorenzen, M., and Schmid, B. (2004). Species evenness and productivity in experimental plant communities. *Oikos*, 107(1):50–63.

Naeini, A. T. and Ghaziasgar, M. (2009). Improving coordination via emergent communication in cooperative multiagent systems: A genetic network programming approach. In *IEEE International Conference on Systems, Man and Cybernetics, 2009. SMC 2009*, pages 589–594.

Nehaniv, C. (2000). The making of meaning in societies: Semiotic and information-theoretic background to the evolution of communication. *Society for the Study of Artificial Intelligence and Adaptive Behavior*, pages 73–84.

Nehaniv, C. L. (2005). Open problems in the emergence and evolution of linguistic communication: A road-map for research. In *Proc. Second International Symposium on the Emergence and Evolution of Linguistic Communication*, pages 86–93.

Nolfi, S. and Mirolli, M. (2009). *Evolution of communication and language in embodied agents*. Springer Science & Business Media.

Parunak, H. V. D., Brueckner, S., Fleischer, M., and Odell, J. (2004). A design taxonomy of multi-agent interactions. *Agent-Oriented Software Engineering IV*, pages 123–137.

Peet, R. K. (1974). The measurement of species diversity. *Annual review of ecology and systematics*, 5:285–307.

Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *SIGGRAPH Computer Graphics*, 21(4):25–34.

Sahraei, B. R., Alers, S., Tuyls, K., and Weiss, G. (2013). Stico in action. In *International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1403–1404, Saint Paul, MN, USA.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, 27.

Sklar, E., Schut, M., Diwold, K., and Parsons, S. (2006). Exploring coordination properties within populations of distributed agents. *AAAI 2006 Spring Symposium on Distributed Plan and Schedule Management*.

Skyrms, B. (2010). *Signals: Evolution, learning, and information*. OUP Oxford.

Spector, L. (2001). Autoconstructive evolution: Push, pushgp, and pushpop. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 137–146.

Spector, L., Klein, J., and Keijzer, M. (2005). The push3 execution stack and the evolution of control. In *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pages 1689–1696. ACM.

Szathmáry, E. (2010). Evolution of language as one of the major evolutionary transitions. In *Evolution of communication and language in embodied agents*, pages 37–53. Springer.

Tuci, E. (2009). An investigation of the evolutionary origin of reciprocal communication using simulated autonomous agents. *Biol. Cybern.*, 101(3):183–199.

Valckenaers, P., Kollingbaum, M., Van Brussel, H., et al. (2004). Multi-agent coordination and control using stigmergy. *Computers in Industry*, 53(1):75–96.

Watson, R. A., Ficici, S. G., and Pollack, J. B. (1999). Embodied evolution: Embodying an evolutionary algorithm in a population of robots. In *Proceedings of the Congress on Evolutionary Computation*, volume 1, pages 335–342, Mayflower Hotel, Washington D.C., USA. IEEE Press.

Weiss, G., editor (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, MA, USA.

Wooldridge, M. (2002). *An Introduction to Multi-Agent Systems*. John Wiley & Sons, Inc., New York, NY, USA.

Artificial life programming in the robust-first attractor

David H. Ackley[1] and Elena S. Ackley[2]

[1]University of New Mexico, Albuquerque, NM 87131
[2]Ackleyshack LLC, Placitas, NM 87043
ackley@cs.unm.edu

Abstract

Despite mounting awareness of the liabilities of deterministic CPU and RAM computing, across industry and academia there remains little clear vision of a fundamental, general-purpose alternative. To obtain *indefinitely scalable computer architectures* offering improved robustness and security, we have advocated a realignment of the roles of hardware and software based on artificial life principles. In this paper we propose an *active media* computational abstraction to underlie such a hardware-software renegotiation. The active media framework is much in the spirit of probabilistic cellular automata, but designed for indefinite scalability and serious programmability, rather than simplicity and analytic tractability. We discuss active media programming techniques based on living systems principles, and present anecdotal data from sample programs to introduce a new programming language called ***ulam***, that we are developing as an underlying language for active media.

Introduction

As the hegemony of CPU and RAM declines, for the first time in decades significantly new computer architectures are appearing—from the nothing-but-net neural architecture of IBM's TrueNorth (Merolla et al., 2014), to the memristor-driven flat parallelism of HP's "The Machine" (Williams, 2014). With the potential on the horizon for a major evolutionary transition in computer architecture, it is an opportune time to reconnect with first principles before shortlisting successors. The result of such a process, we believe, will be the recognition of artificial life as a (perhaps *the*) major force driving future architectural innovation.

Escape from the SDA

Serial deterministic computing based on CPU and RAM is a vast attractor, a valley deep and wide, in a notional space of all possible models of computation. This *Serial Deterministic Attractor* (SDA) is laced with interlocking design decisions surrounding its core demand for logical correctness—which allows the inherent fragility of extremely efficient software to be masked by extremely reliable hardware. Until a bug, or an attacker, appears.

Although the SDA robustness and security properties are dubious, and its scalability is rapidly dwindling, it has been so dominant that alternatives may seem unthinkable. One might imagine that fields like fault tolerance (IEEE, 2013, e.g.,) or probabilistic algorithms (Karp, 1991) fall outside the SDA, but by 'virtually guaranteeing' deterministic execution, they actually entrench it. The same is true of many other non-traditional but still deterministic models, such as synchronous cellular automata (von Neumann and Burks, 1966; Ulam, 1950; Toffoli and Margolus, 1987, e.g.), data flow machines and systolic arrays (Borkar et al., 1988; Budzynowski and Heiser, 2013, e.g.), and asynchronous circuit-level techniques such as GALS and RALA (Kishinevsky et al., 2007; Gershenfeld et al., 2010).

Probabilistic cellular automata (PCA) (Grinstein et al., 1985; Agapie et al., 2014, e.g.) do go decisively beyond determinism, and they are general enough to embrace the kind of models we explore—but their motivations and methods are sharply divergent from the present effort. PCA work often presumes simple and stylized noise models, and proceeds—preferably by formal analysis—to derive insights into equilibrium distributions and other system properties. But when such research begins by postulating a state transition matrix, the small matter of actual PCA programming is silently assumed away. Yes, the transition matrix is a powerfully general device; no, you don't want to program in it.

Recently, there have been some serious programming research efforts that, while remaining mostly traditional, do explicitly abandon determinism and accept some small output errors—often with the motivation of increased parallel efficiency (Cappello et al., 2009; Elliott et al., 2014; Misailovic et al., 2013; Renganarayana et al., 2012, e.g.). We cheer all such efforts but worry they may fail to gain traction because their incremental practicality leaves them struggling up the sides of the SDA valley, with all the downhill directions behind them.

Colonize the RFA

There is at least one fundamental alternative, which we here call the *Robust-First Attractor* (RFA), in the space of all possible models of computation. We have been breaking trail in the RFA for some time (Ackley and Cannon, 2011; Ackley, 2013b; Ackley et al., 2013; Ackley, 2013a; Ackley and Small, 2014a), and can report it is strikingly unlike the SDA, but at least as vast: It is a natural way to understand the computational properties of *living systems*, which have always made do without the luxury of deterministic execution.

Life fills space, as long as suitable resources are available; every RFA architecture must do the same, and that core demand for *indefinite scalability* is surrounded by interacting design decisions often deeply complementary to the SDA's. A von Neumann machine by itself simply isn't an RFA architecture; it is just incomplete, and thus unevaluatable, until a method is defined for tiling unbounded space with it.

Most software-based artificial life models are designed to run on single von Neumann machines.[1] Unsurprisingly, therefore, the properties of such models typically depend critically on deterministic execution, as typified by the utter collapse of constructs in Conway's game of life when facing even mild asynchrony (Bersini and Detours (1994); see also Beer (2014)).

Determinism is a property of the small and the fragile; it is fundamentally misaligned with living systems. It warps our expectations; it's time to move on.

Programmable active media

SDA models are well-suited to implementation in passive, "cold" materials, where uniformity rules, change is rare, and free energy is expensive—conditions where, indeed, living systems may survive but will rarely thrive. However, some environments are diverse in space, dynamic in time, and energetically rich, bountiful, like a rain forest or a sunny day at the shore. We abstract such circumstances into *active media* computational models—unbounded spatial architectures in which each discretized location performs logical state transitions based on its local neighborhood, but with uncertain and variable frequencies and only limited reliability.

An active medium can change spontaneously and is inherently nondeterministic. In a *programmable* active medium we get to pick its state transition function—to specify, up to reliability limits, that certain neighborhood patterns shall stay constant like memory, say, while others produce transitions like a processor or a data transport, or, indeed, act like different types of hardware at different moments. The state transition function we supply is executed asynchronously in parallel across the medium, avoiding overlapping state transitions, again, with good but not guaranteed reliability. It may become possible to implement programmable active media in, say, DNA (Canton et al., 2008; Stojanovic and Stefanovic, 2003, e.g.); it is already possible in electronics (Ackley et al., 2013; Ganapati, 2009).

A new deal for hardware and software

Clearly, compared to an SDA computer architecture, the active media model represents a very different division of labor between hardware and software, as large blocks like 'processor' and 'memory' and 'bus'—and their floorplanning—are placed largely under software control. This refactoring will presumably incur a hardware price-complexity penalty something like FPGA vs ASIC or worse—but that, in turn, may be more than offset by enabling new optimizations akin to RISC vs CISC, combined with the hair-down liberation of merely best-effort hardware determinism.

So, while the programmable active media framework[2] is likely a splendid deal for hardware, it may seem a brutal one-two punch for software, stunned by nondeterminism from below then flattened by expanded mission responsibilities from above. We take that added software engineering complexity as underlying the "hard to program" objection leveled against our approach in a discussion of a very interesting spatial and parallel—though apparently deterministic—model of computation (Budzynowski and Heiser, 2013).

But here's the thing: On the one hand, the software engineering job *should* be harder, because its relative simplicity was purchased with precisely those von Neumann machine features—a single processing locus, uniform passive memory, reliability all on hardware—that led to its Achilles' heels of unscalability and unsecurability. Serial determinism was a simple, sensible starting point, but software engineering and many related fields have emerged since von Neumann's time, and we now know quite a bit about constructing, managing, and evolving complex systems. Looking back from the RFA, for software still to be demanding general pointers and flat RAM and cache coherent global determinism seems like clutching blankie. The future will arrive anyway.

That said, and on the other hand, software's big promotion becomes less terrifying as we get down to work, because, like hope from Pandora's box, "best effort" wafts upwards from the nondeterministic hardware into the software as well. As a system component, we'll do our best with what we've got and what we get, but if things go really wrong, we can simply delete ourselves and let our kin cover for us. Correctness and robustness are measured by degrees and circumstances in living systems; in the RFA they are highly respected qualities rather than merely purported necessities.

Outline

The rest of this paper focuses on the challenges and opportunities of indefinitely scalable programming in the RFA. The next section briefly presents a few slogans or rules of

[1] Though there have certainly been exceptions, both proposed (Ray, 1995) and implemented (Ackley, 1996). Additionally, powerful modeling systems like *Ready* (Hutton et al., 2012), though still fundamentally deterministic, now exploit many-core parallelism.

[2] It's probably too late to call it "Software-Defined Hardware"?

thumb we have developed, to serve both as antidotes to decades of SDA thinking and as prototype design patterns, yet to be fully detailed, for RFA programming based on living systems principles. Then Section "*The Ulam programming language*" briefly introduces our new language, Section "*Programming examples on active media*" presents several examples, and finally Section "*Artificial life on active media*" offers brief concluding thoughts.

Principles of active media programming

To help make the RFA more concrete and clearly distinct from the SDA, in this section we offer some RFA slogans or design principles, with brief alife motivations or implications. The end of Section "*A new deal for hardware and software*" above concerns a principle that might be called *There's always dying*; here are five more:

- *Happy and you know it*: Make the goals obvious,
- *Space is the place*: Space is the core data structure,
- *Embrace the race*: Just help the better answer win,
- *Chance it*: Replace state with statistics, and
- *Use it or lose it*: A cycle saved is a cycle wasted.

Happy and you know it: The key to robustness is effective redundancy, and one excellent approach is to give subcomponents not just tasks to do but also ways to measure their own success. Unit tests are a simple SDA example; in the RFA, geometric goals ("Bigger should be lower") and local consistency rules ("I should be able to see you seeing me") allow the execution of external 'productive' computations to be combined with ongoing internal processes like machine construction and maintenance.

Space is the place The SDA tries to obliterate space using Random-Access Memory, then acts all surprised by buffer overflows; the RFA uses space as the backbone organizing principle for both access control and computation, like the cat that's picky about who's allowed near, but then grooms everything in reach. We are in largely good shape here: A great strength of typical alife models is their fundamentally spatial organization, unlike, say, typical genetic algorithms.

Embrace the race The SDA generally abhors race conditions[3] but in the RFA, with many components interacting and making things better locally, a race can be, not a regrettable shame to be hidden, but a fine tool for making a larger-scale decision, which may then be amplified by spectators for subsequent processing. Alife models also do well with this—often identifying its race conditions with names involving "selection".

Chance it The SDA is typically deterministic even when it's not supposed to matter, as in breaking ties or sizing a supposedly ample buffer or timeout. But with determinism off the table many RFA tasks can be dramatically simplified by replacing state with probabilities, and rather complex dynamics can be implemented using remarkably little per-object state, as demonstrated in Section "*Stochastic timer*" below. Alife models are mixed on this, but the field faces some hard unlearning.

Use it or lose it The active media abstraction is a programmable space in which energy is a nonrefundable sunk cost, and even though actual power is limited, this idealization shifts the discussion from the stultifying task of minimizing the cost of energy consumed to the galvanizing task of maximizing the value of energy provided. Other things being equal, the RFA programmer seeks to minimize the *average change age per site* in some productive way.

Such slogans can help establish shared context, but code is what really talks, and that's where we head next.

The Ulam programming language

For some time, we have been exploring an indefinitely scalable architecture called the *Movable Feast Machine* (MFM) (Ackley and Cannon, 2011; Ackley et al., 2013; Ackley, 2013a). We have recently developed an open-source C++ MFM implementation (Ackley and Small, 2014b) that is currently incorporated into thread-per-tile simulators running on conventional multicores, but is designed and written for eventual cross-compilation onto indefinitely scalable "tile per tile" physical hardware.

We have also been developing an open-source compiler for a new language we call ***ulam*** (Ackley and Ackley, 2014), mostly named after Stanislaw Ulam for his pioneering contributions especially in cellular automata (Ulam, 1950), but also for various weak bacronyms such as that concluding this paper's abstract. We give only a few bullet points here, for grounding and flavor—there's runnable code in the next section—but ***ulam*** deliberately looks, and is, rather like a conventional object-oriented procedural language:

- The top-level compilation driver, called `ulam`, is a Perl script; `culam`, the ***ulam*** compiler proper, is written in C++, and its output is heavily templated C++, so that despite weird data sizes and packing (see next point), the `g++` compiler downstream can sometimes find quite delicious assembly code sequences.
- Numbers can be declared in any width from 1 to 32 bits, using both more and less conventional interpretations (Table 1). All primitive type casts and assignments saturate in the destination type. One dimensional arrays of zero or more items are provided, bit-packed up to 32 total bits.
- The ***ulam*** analog to an object instance is called an `Atom`, but an `Atom` is also analogous to a word of memory in a

[3] Even those rare authors who accept non-determinism usually see races as no more than tolerable (Misailovic et al., 2013; Boehm, 2012), although there are exceptions (Madhavan et al., 2014, e.g.).

Ulam type	*Interpretation*
`Unary(`k`)`	Base 1 (population count)
`Unsigned(`k`)`	Base 2 positional
`Int(`k`)`	Two's complement base 2
`Bool(`k`)`	Boolean (majority of pop. count)
`Bits(`k`)`	Uninterpreted bit values

Table 1: *ulam* primitive bit structures, widths $k = 1..32$ (except `Bool` uses only odd widths).

tagged architecture, so each one is the same size—96 bits in the current design, with 25 reserved for the object type and error correction. The analog to a class is called an `element`, which may define (non-static) methods and up to (96-25=) 71 bits of data members. A `native` keyword instead of a method body provides an escape to C++, with all the usual caveats.

- There is composition but no inheritance; the analog to a struct is called a `quark`, which may be any size from 0 to 32 bits. There is also `union` with the typical meaning. The `has` operator determines if an `Atom` contains a given quark, so a quark method (for example) can try to 'find one of its own' inside an arbitrary atom.
- For indefinite scalability, the only persistent data memory is a spatial grid of `Atom` storage; there is no dedicated heap or random-access main memory. There is a per-event function call stack, and an `EventWindow` library provides read/write access to the local grid neighborhood, comprising 41 total `Atom`s within Manhattan distance 4 of the center. (The underlying MFM tile structure is not directly accessible from ***ulam***.)
- All method arguments are passed by value, except the implicit `self` argument is passed by reference. There is no privileged `main()` method; from the point of view of an element, execution occurs during a call to a handful of special methods (see Table 2), especially the `behave` method, which is called to perform one state transition on the `Atom` and/or its encompassing `EventWindow`.

Special method	*On*	*Purpose*
`Void behave()`	*E*	Perform event
`Int test()`	*EQ*	Run unit tests
`Int toInt()`	*Q*	Custom cast to `Int`
`T aref(Int)`	*EQ*	Custom array read
`Void aset(Int, T)`	*EQ*	Custom array write

Table 2: ***ulam*** privileged methods for elements (*E*) and/or quarks and unions (*Q*)

```
1 ulam 1;
2 /** Fork bomb.
3   \symbol FB \color #f00
4   \symmetries all
5 */
6 element ForkBomb {
7   EventWindow ew;
8   Void behave() { ew[1] = ew[0]; }
9 }
```

Figure 1: A complete ***ulam*** element. Copies itself from the event window center (`ew[0]`) to `ew[1]`, which in this case might be any adjacent site. See text.

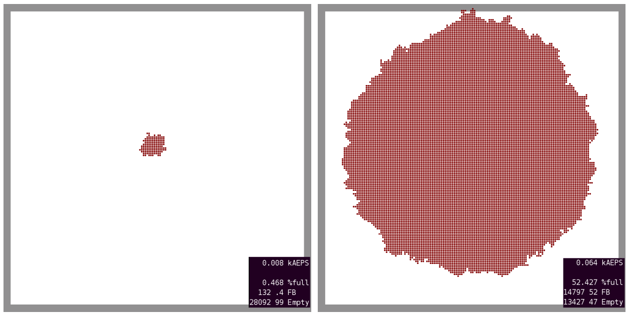


Figure 2: Uncontrolled (and uncontested) growth of a single initial `ForkBomb`. *Left:* After an average of 8 events per site (8 AEPS). *Right:* After 64 AEPS.

Programming examples on active media

Here, briefly presented, are five examples of ***ulam*** code that we have created recently, embodying living principles like reproduction (uncontrolled and controlled) and group formation and action. In a nod to more traditional engineering, rather than only alife, the final example is a tiny toy data switch implementation.

Fork bomb

Figure 1 presents complete ***ulam*** code for a simple runaway-reproducing fork bomb. *Line 1* declares the language version in use. *Line 7* declares an instance of the `EventWindow` library, as a size 0 data member. In addition to the expected 2D neighborhood access methods, `EventWindow` also offers a one-dimensional mechanism that often suffices, as here: The `self`—the atom having the event—is by definition at `ew[0]`, with increasing 1D array indices spreading outwards in 2D breadth-first.

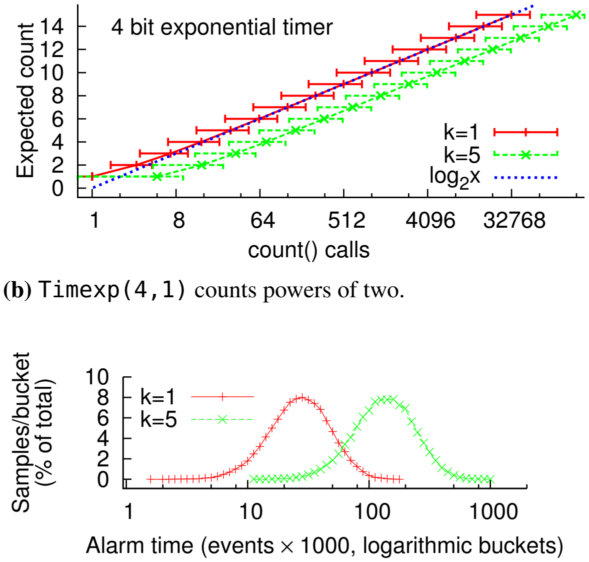
With up as "north", `ew[1]` is the adjacent site east, so one might expect `ForkBomb` to produce an eastward growing line. But the element metadata (inside the structured comment, at *Line 4*) declares that all eight square rotations and reflections are valid for `ForkBomb`, so a random transform is chosen for each of its events, and it explodes (Figure 2).

Running the `ulam` driver on `ForkBomb.ulam` compiles

```
ulam 1;

/** A stochastic exponential timer; a template taking
  an exponential factor (exp) and a multiplicative
  factor (k). After a reset(), count() can be called
  approximately k*2**exp times before alarm() will
  begin returning true.
*/
quark Timexp(Unsigned exp, Unsigned k) {
  Random r; // PRNG is infrastructure, costs 0 bits
  Unsigned(exp) t; // costs exp bits
  Void reset() { t = 0; }
  Unsigned count() {
    if (!alarm() && r.oneIn(k<<t))
      ++t; // Each tick about doubles in length
    return t;
  }
  Bool alarm() { return t == t.maxof; }
}
```

(a) Complete ***ulam*** code of a quark template.



(b) `Timexp(4,1)` counts powers of two.

(c) The alarm time distributions of `Timexp(4,k)` are symmetric on the logarithmic scale.

Figure 4: A stochastic timer. A four-bit `Timexp(4,1)` separates time scales over four orders of magnitude, combining fine-grained initial resolution with overall long duration.

```
ulam 1;
quark Telomere(Unsigned width) {
  typedef Unsigned(width) Tail;
  typedef EventWindow.SiteNum SiteNum;

  EventWindow ew;
  Tail age;
  /** Duplicate into ew[to] if self isn't
    too old and ew[to] is a live, empty site. */
  Bool dup(SiteNum to) {
    if (age < Tail.maxof) {
      if (!ew.isLive(to) || !(ew[to] is Empty))
        return false;
      ++age; // Must increment before copying!
      ew[to] = self;
    }
    return true;
  }
}
```

Figure 3: An excerpted `Telomere.ulam`, offering a quark that provides controlled growth to elements containing it.

it into C++ 'intermediate code', which is then processed by the standard **gcc** tools, yielding a custom ***ulam*** element library (`libcue.so`) that can be dynamically loaded into the Movable Feast Machine simulator—and, hopefully soon, into actual hardware tiles. Loaded into the simulator, the library adds the `FB` element to the available element palette, allowing computations like Figure 2 to be run immediately.

Telomere

The forkbomb is so simple and obvious that it's sort of a "Hello world!" for active media, but its cancerous rampage is a poor example of alife programming. Inspired by the *telomere* DNA sequences that shrink during reproduction, Figure 3 illustrates a reusable ***ulam*** software component that provides controlled reproduction of a single starting cell into a clonal population of $2^{2^{width-1}}$ atoms, assuming all clones survive and spread out sufficiently. The group-forming element in Section "*Mob rule*", below, uses this strategy.

Stochastic timer

Figure 4a shows an example of using statistics instead of state as mentioned with the "Chance It" principle in Section "*Principles of active media programming*". `Timexp` is a quark template taking a bit size (`exp`) parameter, and a multiplier (`k`). The `exp` specifies both its range of counting ability, and how much it consumes from the atomic bit budget; `k` scales the entire curve.

So a `Timexp(4,1)` quark, for example, can be dropped into any element with four bits to spare, and its `count()` method can be called thousands of times, with very high probability (Figure 4b), before its `alarm()` method returns `true`. When its `t` data member is 0, `k<<t` is 1 and so `oneIn()` returns `true` for sure, but each successive increment doubles the odds and on average takes twice as long.

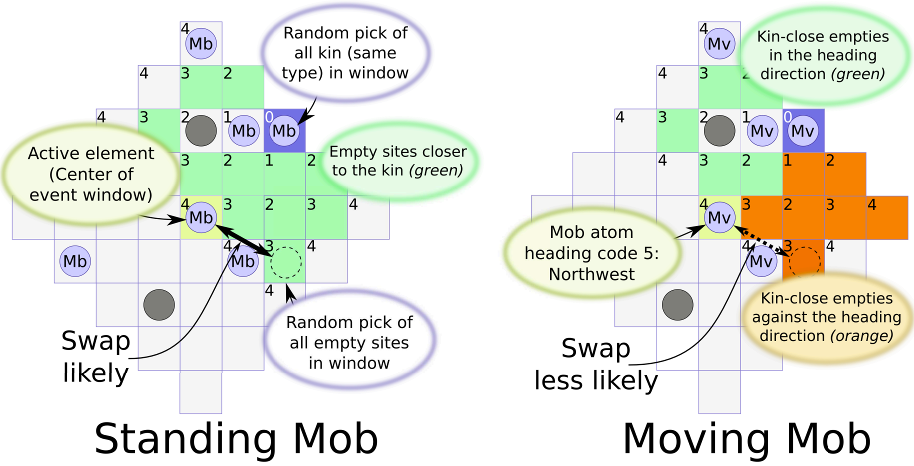
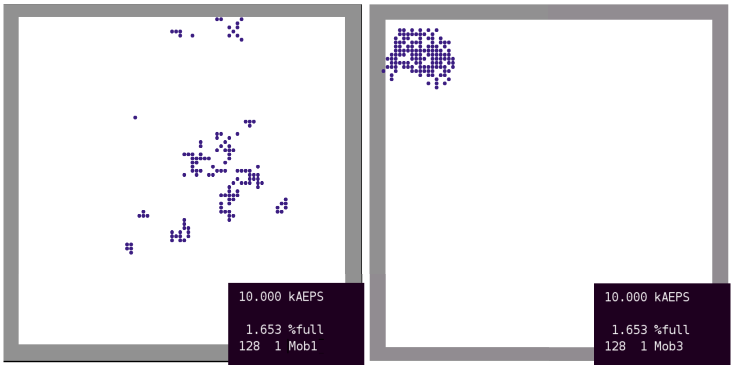

Random pick of all kin (same type) in window
Active element (Center of event window)
Empty sites closer to the kin (green)
Random pick of all empty sites in window
Swap likely
Mb
Standing Mob
Kin-close empties in the heading direction (green)
Mob atom heading code 5: Northwest
Kin-close empties against the heading direction (orange)
Swap less likely
Mv
Moving Mob

Figure 5: Two mob atom types evaluate their options, having picked at random from the empty sites (*green* or *light grey*) and a random other mob atom. *(Left)* The basic mob rule: Swap if the empty is closer to the kin. *(Right)* Mobs induce drift by discouraging moves against a heading direction (*orange* or *dark grey*). See text and Figure 6.


```
10.000 kAEPS

 1.653 %full
128  1 Mob1
```

```
10.000 kAEPS

 1.653 %full
128  1 Mob3
```

Figure 6: *(Left)* At 10 kAEPS a mob using just the basic mob rule (Figure 5, left) has broken up, but *(right)* adding statistical gravity and a directional bias (Figure 5, right) yields a long-lived, slowly-moving mob. Mobs began as single centered atoms.

Mob rule

This fourth example demonstrates both cohesion and mobility in a single structure—a "mob"—that's larger than an event window but smaller than the whole universe. A `Telomere(3)` quark induces a single "Mob" atom to form a cluster of 128 clones all following the *mob rule* (Figure 5). That proved insufficient for mob cohesion (Figure 6, left), but it works if we skip some mob moves with a probability that grows with the number of `Mob` atoms in our event window, a fix we call *statistical gravity*. (Figure 6, right).

Mobs are new but we hope they find use—as either code or inspiration—as a medium-scale mobile spatial data structure. They are slow and sloppy, but note all that dynamics—the managed growth, the group cohesion, and the controllable, directed movement—costs just one byte per atom.

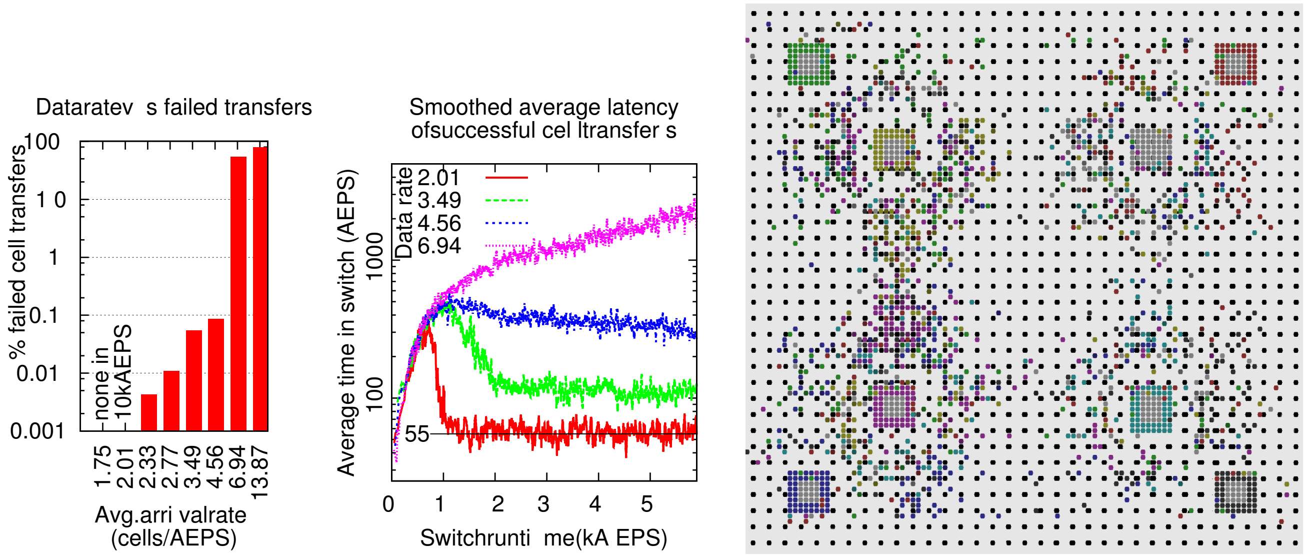
Data switch

As a final example, to explore and stress ***ulam***'s programmability, we developed a toy "data switch", designed to carry fixed-size data cells between eight bidirectional ports. We made `Router` atoms that self-assemble into a grid, while gossiping amongst themselves to derive spatial gradients towards eight `Port` atom clusters, which emit and consume gradient-following data `Cell` atoms, each carrying a four byte payload destined for a random other `Port`.

In this demo, about 550 lines of switch-specific ***ulam*** code compile into about 30K lines of very stylized C++; Figure 7 shows some data (*7a*, *7b*) and a day-in-the-life image redrawn from a screenshot (*7c*). This little switch knows nothing of packet reassembly and sometimes drops cells, and it's slow, jittery, prone to catastrophic crowding at higher data rates, totally impractical and completely glorious.

Artificial life on active media

Active media are powerful but twitchy. Depending on their programming, small deviations may grow, positive feedback loops may erupt, chain reactions are possible. Of course there are fork bombs in traditional computing as well, but



(a) Cell arrival rate vs % lost. **(b)** Data switch latency vs machine uptime. **(c)** Snapshot at 10kAEPS; cell arrival rate=4.56.

Figure 7: Performance of the 8 port switch: The data loss *(a)* is low until a critical arrival rate is reached; initial average latency *(b)* is high until the routing grid is constructed. Colored square borders in *(c)* are `Port` atoms, some in the color of just-received data; scattered dots are in-flight data `Cells`; the grid of `Router` atoms can be seen against the grey background. See text.

indefinite scalability forces the issues by raising the stakes; that is a feature.

Yes, to a degree, programming active media genuinely will be harder, but that's because the programs, once written, will do that much more for us, providing a vertical slice from function to floorplan to hardware construction and maintenance. And partly, the programming will just *seem* harder because we're unfamiliar with its idioms, and we haven't done the software engineering yet, so the accommodations look quite rustic. But as suggested by the work reported in this paper, that last problem we are starting to fix.

The ***ulam*** compiler has been under heavy development since August 2014, and to coincide with ECAL 2015 we are planning an official version 1.0 release, complete with documentation, tutorials and Ubuntu packaging so that installation can be as simple as `apt-get install ulam` from a public personal package archive.

There is so much to be learned and relearned, designed and redesigned, implemented and reimplemented. It can seem positively daunting, but we hope to entice you or your students (or advisors!) to give it a try.

You might be a Ruby fan, or Python or Haskell or Java or Forth; some if not many of our language design decisions in ***ulam*** will almost surely not be your cup of tea; that's okay. We just need to keep our eyes on the prize: The goal is neither maximum parallel efficiency, nor maximum expressive purity. The goal is *indefinite scalability while balancing concurrency and programmability*—and to that end, robustness must be inherent not just in hardware, but across the computational stack.

We must explore and colonize and settle the RFA, but ultimately we will also leverage the technology that is now filling our society, after seventy years of relentless optimization in the serial deterministic attractor. There is nothing wrong with von Neumann machines that cannot be fixed by making them small and individually insignificant parts of an indefinitely scalable architecture.

Change is coming. Alife research can lead the way.

Acknowledgments

This work was supported in part by a Google Faculty Research Award, and in part by grant VSUNM201401 from VanDyke Software, both to the first author. Trenton R. Small implemented much of the C++ MFM implementation. Special thanks to the F'14 "Robust Artificial Life" students at UNM for contributing ideas and enthusiasm, and for exploring lots of hand-coded C++ elements for their projects.

References

Ackley, D. H. (1996). ccr: A network of worlds for research. In Langton, C. and Shimohara, K., editors, *Artificial Life V. (Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems)*, pages 116–123, Cambridge, MA. The MIT Press.

Ackley, D. H. (2013a). Bespoke physics for living technology. *Artificial Life*, 19(3_4):347–364.

Ackley, D. H. (2013b). Beyond efficiency. *Commun. ACM*, 56(10):38–40. Author preprint: `http://nm8.us/1`.

Ackley, D. H. and Cannon, D. C. (2011). Pursue robust indefinite scalability. In *Proc. HotOS XIII*, Napa Valley, California, USA. USENIX Association.

Ackley, D. H., Cannon, D. C., and Williams, L. R. (2013). A movable architecture for robust spatial computing. *The Computer Journal*, 56(12):1450–1468.

Ackley, D. H. and Small, T. R. (2014a). Indefinitely scalable computing = artificial life engineering. In *Proceedings of The Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14) 2014*, pages 606–613. MIT Press.

Ackley, D. H. and Small, T. R. (2014b). The MFM version 2 codebase. `https://github.com/DaveAckley/MFMv2`.

Ackley, E. S. and Ackley, D. H. (2014). The Ulam compiler for MFM programming language. `https://github.com/elenasa/ULAM`.

Agapie, A., Andreica, A., and Giuclea, M. (2014). Probabilistic cellular automata. *Journal of Computational Biology*, 21(9):699–708.

Beer, R. D. (2014). The cognitive domain of a glider in the game of life. *Artificial Life*, 20(2):183–206.

Bersini, H. and Detours, V. (1994). Asynchrony induces stability in cellular automata based models. *Artificial Life IV*, pages 382–387.

Boehm, H.-J. (2012). Position paper: Nondeterminism is unavoidable, but data races are pure evil. In *Proceedings of the 2012 ACM Workshop on Relaxing Synchronization for Multicore and Manycore Scalability*, RACES '12, pages 9–14, New York, NY, USA. ACM.

Borkar, S., Cohn, R., Cox, G., Gleason, S., Gross, T., Kung, H. T., Lam, M., Moore, B., Peterson, C., Pieper, J., Rankin, L., Tseng, P. S., Sutton, J., Urbanski, J., and Webb, J. (1988). iWarp: An integrated solution to high-speed parallel computing. In *Proceedings of Supercomputing '88*, pages 330–339.

Budzynowski, A. and Heiser, G. (2013). The Von Neumann architecture is due for retirement. In *Proceedings of the 14th USENIX Conference on Hot Topics in Operating Systems*, HotOS'13, pages 25–25, Berkeley, CA, USA. USENIX Association.

Canton, B., Labno, A., and Endy, D. (2008). Refinement and standardization of synthetic biological parts and devices. *Nature biotechnology*, 26(7):787–793.

Cappello, F., Geist, A., Gropp, B., Kal, L. V., Kramer, B., and Snir, M. (2009). Toward exascale resilience. *IJHPCA*, 23(4):374–388.

Elliott, J., Hoemmen, M., and Mueller, F. (2014). Exploiting data representation for fault tolerance. In *Proceedings of the 5th Workshop on Latest Advances in Scalable Algorithms for Large-Scale Systems*, ScalA '14, pages 9–16, Piscataway, NJ, USA. IEEE Press.

Ganapati, P. (2009). Hardware hackers create a modular motherboard. `http://www.wired.com/gadgetlab/2009/08/modular-motherboard`.

Gershenfeld, N., Dalrymple, D., Chen, K., Knaian, A., Green, F., Demaine, E. D., Greenwald, S., and Schmidt-Nielsen, P. (2010). Reconfigurable asynchronous logic automata: RALA. In *Proceedings of the 37th annual ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, POPL '10, pages 1–6, New York, NY, USA. ACM.

Grinstein, G., Jayaprakash, C., and He, Y. (1985). Statistical mechanics of probabilistic cellular automata. *Phys. Rev. Lett.*, 55:2527–2530.

Hutton, T., Munafo, R., Trevorrow, A., Rokicki, T., and Wills, D. (2012). Ready, a cross-platform implementation of various reaction-diffusion systems. `https://code.google.com/p/reaction-diffusion`. Accessed Mar 2015.

IEEE, editor (2013). *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Budapest, Hungary, June 24-27, 2013*. IEEE.

Karp, R. M. (1991). An introduction to randomized algorithms. *Discrete Applied Mathematics*, 34(13):165 – 201.

Kishinevsky, M., Shukla, S. K., and Stevens, K. S. (2007). Guest editors' introduction: GALS design and validation. *IEEE Design and Test of Computers*, 24:414–416.

Madhavan, A., Sherwood, T., and Strukov, D. (2014). Race logic: a hardware acceleration for dynamic programming algorithms. In *Computer Architecture (ISCA), 2014 ACM/IEEE 41st International Symposium on*, pages 517–528. IEEE.

Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., Jackson, B. L., Imam, N., Guo, C., Nakamura, Y., Brezzo, B., Vo, I., Esser, S. K., Appuswamy, R., Taba, B., Amir, A., Flickner, M. D., Risk, W. P., Manohar, R., and Modha, D. S. (2014). A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673.

Misailovic, S., Kim, D., and Rinard, M. (2013). Parallelizing sequential programs with statistical accuracy tests. *ACM Trans. Embed. Comput. Syst.*, 12(2s):88:1–88:26.

Ray, T. S. (1995). A proposal to create a network-wide biodiversity reserve for digital organisms. Technical Report Technical Report TR-H-133, ATR.

Renganarayana, L., Srinivasan, V., Nair, R., and Prener, D. (2012). Programming with relaxed synchronization. In *Proceedings of the 2012 ACM Workshop on Relaxing Synchronization for Multicore and Manycore Scalability*, RACES '12, pages 41–50, New York, NY, USA. ACM.

Stojanovic, M. N. and Stefanovic, D. (2003). A deoxyribozyme-based molecular automaton. *Nature Biotechnology*, 21(9):1069–1074.

Toffoli, T. and Margolus, N. (1987). *Cellular Automata Machines: A New Environment for Modeling (Scientific Computation)*. The MIT Press.

Ulam, S. (1950). Statistical mechanics of cellular automata, 1952. *Proceedings of the International Congress on Mathematics*, 2:264–275.

von Neumann, J. and Burks, A. W., editors (1966). *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana, IL, USA.

Williams, S. (2014). K2I distinguished lecture - The Machine: The HP memristor solution for computing big data. Video retrieved January 2015 from `https://mediacosmos.rice.edu/app/plugin/embed.aspx?ID=vVUTzFCE006yZu3TF1sXKg&destinationID=URka-_E5-0-e4girH4pDPQ&contentID=vq2oQtAqPkeAqm5F6uSnqA`.

Towards a methodology for describing the relationship between simulation and reality

Eric Schneider[1], Elizabeth I. Sklar[1], M. Q. Azhar[2], Simon Parsons[1] and Karl Tuyls[1]

[1]Department of Computer Science, University of Liverpool, UK
{eric.schneider, e.i.sklar, s.d.parsons, k.tuyls}@liverpool.ac.uk
[2]Borough of Manhattan Community College and Graduate Center,
City University of New York, New York, USA
mazhar@bmcc.cuny.edu

Abstract

For research that carries out experiments in simulation, an important question is how the results will translate into the real world. This paper proposes a method for comparing results obtained in simulated versus physical environments, based on *interval* relationships between metrics gathered in both settings. The approach is motivated by the fact that the relationship between absolute measures often does not tell much. For example, the amount of time taken to complete a task in simulation versus the same task in the physical world could always be shorter in simulation because of speed-up factors embedded in the simulator. Three different metrics are introduced that describe different interval relations, and these are demonstrated using two case studies.

Introduction

It is common practice in *artificial life*, *evolutionary computation*, *multiagent systems* and *robotics* to employ *simulation* as a means to evaluate an approach which is intended to be deployed in some type of *real* environment. "Real" might be physical (as in the case of robotics) or might be interactive (as in the case of human-agent systems) or might be real-time (as in the case of systems that respond to live data feeds, such as financial stock prices, or sensors, such as traffic lights). The advantage of simulation over reality is that we typically have more control over and easier access to the simulated environment. This implies that it is simpler to test algorithms, or whatever we are working on, in the simulated environment first—i.e., before it is deployed in reality.

Notwithstanding the many issues in transferring results from simulation to reality (Brooks (1992)), the general wisdom is that "if it works in simulation", then it will work, to some degree, in reality; and if it doesn't work in simulation, then it certainly will not work in reality. While there is a reasonable literature on the notion of *verification*, especially in multiagent systems, and some attention paid to the notion of *validation* in multiagent-based simulation (though not enough, in our opinion), these pieces of work do not attempt to measure **how good** a simulation is with respect to the reality it is meant to approximate and and **how well** the simulation solution will **transfer** to reality. In other words, if we say that testing in simulation will ensure that a particular approach will work in reality *to some degree*, what does that mean? To *what* degree? And what is a *degree*?

In the work presented here, we address exactly those questions. Our contention is that a simulation environment will never fully or exactly emulate everything that happens in a real environment (agreeing with Brooks (1992)), but if we have some structured way of measuring and describing what the degree of closeness is, then we will have a structured way of being able to define how robust our approach—tested in simulation—is with respect to reality. Especially in cases where testing in reality is risky (e.g., nuclear cleanup) and/or expensive (e.g., planetary exploration), it would be very useful to know how much we are gaining by the knowledge obtained in the simulation environment. This work is particularly relevant in the *artificial life* community because our methodology can be applied to assess the utility of *artificial* approximations or imitations of real phenomena.

This paper is organised as follows. The next section highlights prior work on describing the relationship between real and simulated environments. Then we describe our methodology in abstract terms, and outline an example in which we applied our methodology to some of our own work in robotics and multiagent systems. Finally, we close with some discussion and conclusions.

Related Work

As it is often infeasible to develop robot behaviours on physical hardware, there has been a good deal of investigation into developing behaviours in simulations. An early example of this is Koza (1991), which used genetic programming to recreate the kind of navigation that Mataríc (1990) had hand-coded, and led some to conclude that it would be straightforward to use evolutionary techniques to learn robot controllers that could be dropped into real robots that would then operate as desired in the real world.

Responding to this position, Brooks (1992) raised concerns about the transferability of behaviours learned in simulations due to significant differences between simulation and physical environments.

First, working purely in simulation, that is without regularly checking the results of the simulation against what happens in the real world, could lead to evolutionary techniques focusing on problems that just don't exist in the real world. Second, if simulators do not accurately model the errors that occur in sensing and actuation, evolutionary techniques that evaluate their output only in simulation are unlikely to evolve controllers that will work on real robots.

Jakobi et al. (1995) introduced the term *reality gap* to describe the differences between reality and simulation that Brooks had described, and went on to provide evidence both of the existence of the gap and of the possibility of overcoming it. They evolved controllers under three conditions: no noise, noise equivalent to that measured in the real world ("observed noise" in their terminology), and much more noise that is observed in reality. Controllers evolved with observed noise worked when transferred onto a real robot. Controllers evolved either with no noise, or with much more than observed noise, failed on real robots.

There have been efforts to skirt the reality gap rather than model it explicitly. Vaughan and Zuluaga (2006) propose using a simulator at various points during the performance of a physical task—selecting targets and planning paths to them—in order to find viable solutions, and especially to avoid dangerous outcomes like colliding with walls or other obstacles. This use of simulation is similar to real-time planning methods like the Dynamic Window Approach (Fox et al., 1997) to collision avoidance. More recently, Farchy et al. (2013) used "Grounded Simulation Learning" to optimize a walk cycle on a humanoid robot. "Grounding" involved learning a controller via a small number of trials on a physical robot before refining the controller through a much larger number of trials in a simulator. Further development occurred over a number of round trips through this process. Rather than tune the simulator to match observed noise in the robot's physical environment, the *behaviour* of the simulated robot was constrained to match real world results.

Marques and Holland (2009) define architectures for "functional imagination" for simulation, that is, architectures in which behaviours developed in simulation are transferable to physical implementations in some useful way. They identify a set of necessary and sufficient features for a simulator to provide "behavioural benefit" to physical performance, but do not directly address the problems raised by the reality gap, much less how to measure or overcome it.

Koos et al. (2010) note that transferability and efficient performance in a simulator, which may exploit bugs or poor models of a physical environment, are conflicting goals. They propose an evolutionary algorithm that aims to optimize for both objectives. To help achieve this, they define a "simulation-to-reality disparity factor" for controllers developed in simulation. This factor is based on the differences in the controller's performance observed in simulation and physical environments according to certain "behavioural features". Examples of features are distance covered during an experiment or the angular orientation of a robot and the end of its behaviour.

Approach

The question of how well simulation predicts performance in a physical environment can be examined in a number of ways. A simple method is to select a particular metric, e.g., *distance travelled* in a mobile robot domain, and compute that measurement in the robot's physical environment as well as in a simulated environment with the same geometric specifications. In earlier work (Sklar et al., 2012), we did just this. We selected six metrics and ran point-to-point comparisons between the physical environment and a parallel simulated environment. Our results showed that, while the individual metrics—scalar values, i.e., single *points* within a distribution—do not line up in absolute terms, they do align in relative terms. We had the idea that the *relationships* between metrics could be expressed as some *function* that could be computed from sample results collected in physical and simulated environments. In this way, one could collect a statistically significant sample in simulation, and then apply the function to those results and obtain a fair approximation of what the physical results would be. For example, we could train a neural network to predict the physical results based on simulated results, using our sample data set as the training set.

Here, we continue this line of inquiry, but propose three additional ways of looking at the relationships between simulated and physical results:

- First, instead of performing point-to-point comparisons, we examine the distribution of values for a particular metric and compute an *interval* that describes the bounds of the distribution (such as $\mu \pm \sigma$ or $max - min$ value range). We can then compare the intervals for individual metrics, in simulation versus physical environments.
- Second, in addition to considering intervals for individual metrics, we compare the intervals for *sets* of metrics.
- Third, we consider *rank order* comparisons of groups of metrics. Since simulation is often used to assess the impact of various experimental conditions, we hypothesize that the best-to-worst ordering of each condition can be compared in physical and simulated environments and the ordering itself can be useful even if the point-to-point or interval relationships do not align.

We believe that these comparisons are useful additions to the point-to-point comparisons in describing the predictive power of a simulation environment with respect to a parallel physical environment. For example, rather than saying that simulation results predict physical results "to some degree", we can say that the results are comparable with respect to

relations		*picture*
$x < y$	$y < x$	`XXX YYY`
$x = y$		`XXX` `YYY`
x meets y	y meets x	`XXXYYY`

x overlaps y	y overlaps x	`XXX`
		` YYY`

x during y	y during x	` XXX`
		`YYYYYY`

x starts y	y starts x	`XXX` `YYYYY`

x finishes y	y finishes x	`  XXX`
		`YYYYY`

Figure 1: Allen's 13 temporal interval relationships between two time periods, x and y (from Allen (1983)).

statistical intervals or *rank ordering*. Next, we describe each methodology in detail.

Statistical interval comparison

Allen (1981, 1983) describes temporal relations between events and identifies thirteen possible relationships between any pair of time intervals. These are listed in Figure 1.

We apply the same idea to any scalar metric that can be expressed as a *statistical interval*. For example, it could be a *confidence interval*, centered on the mean and bounded by $\pm n$ standard deviations; or it could be a *quartile* interval; or a *min-max* interval. Applying the statistical interval relationship method works as follows:

1. Collect a set of raw data for a particular metric in the simulated environment—a statistically significant sample—and another set of data for the same metric in the corresponding physical environment (a smaller sample).

2. Compute the mean, μ, and standard deviation, σ, of both samples (note that we assume that the distribution in the physical environment will be normal, thus the mean and standard deviation are still valid, albeit not as reliable as when the data set is larger).

3. Plot the interval $[\mu - n\sigma, \mu + n\sigma]$ for both simulated and physical data sets, as two vertical columns in a 2-dimensional graph. We select n based on the percentage of the distribution that we want our interval to cover. For example, $n = 1$ will cover 68% of the distribution and $n = 2$ will cover 95% of the distribution[1].

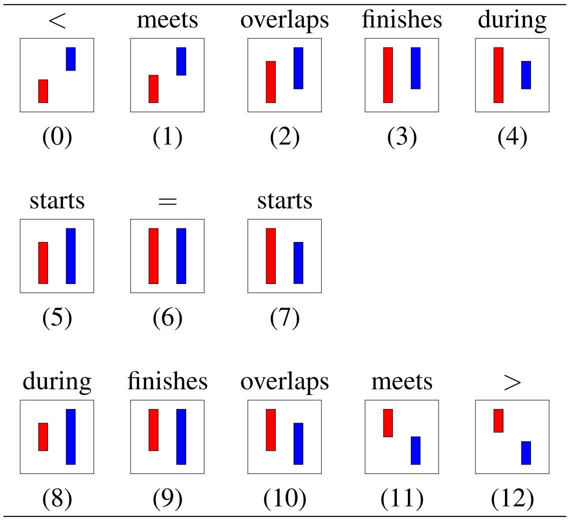


Figure 2: Statistical intervals, using Allen's relations labels. See text for explanation.

Now we can examine the relationships between these intervals, as illustrated in Figure 2. The relations are labelled using Allen's terminology, but the order in which the relations are displayed is sorted so that the equality relationship is in the middle and the further we go from the middle, the more disparity between the values being compared. Cases (0) and (12) are when the data is completely unaligned and one set of values is strictly less (or greater) than the other. Cases (1) and (11) are when one set of values is less (greater) than or equal to the other. Cases (2) and (10) are when the two sets of values overlap. Cases (3) and (9) are when the upper bounds of both sets are equal, but not the lower bounds. Cases (4) and (8) are when one set of values is completely contained in the other set. Cases (5) and (7) are when the lower bounds of both sets are equal, but not the upper bounds. Case (6) is when the data is perfectly aligned.

We use the statistical intervals to compare experimental results under two different conditions. These could be *physical* versus *simulated*, which is what we are particularly interested in here; however, these could generalise to comparing other pairs of experimental conditions. For example, consider Case (8). This case indicates that the performance in one metric (blue) "contains" the performance of the other (red). This implies that performance under the blue condition is more variable than under the red condition. If we are comparing performance in, say, simulation in the left-hand (red) interval and physical in the right (blue), we might be able to say that performance (in this metric) was more variable in the physical setting. If our goal is to derive behaviours in simulation (red) that are guaranteed to fall within

[1] These are standard values for normal distributions.

a certain interval in the physical world (blue), then we cannot make this claim if the data matches Case (8); however, if our data (red:simulation; blue:physical) matches Case (4), then we can make the claim.

Alternatively, we might be interested in comparing metrics resulting from experimental conditions that differ in ways other than simulated versus physical. For example, we might be interested in comparing how a robot interacts with a person when the robot is programmed using two different behaviours, called `beh1` and `beh2`. Experimental results could be obtained from people interacting with a physical robot that exhibits both behaviours, as well as with a simulated robot that also exhibits both behaviours. If the statistical interval relationship for one metric resulting from `beh1` (red) compared with `beh2` (blue) falls into Case (2) for the physical robot, and does the same for the simulated robot, then we can be confident that the simulation environment produces results reliably similar to the physical environment in order to be able to use the simulated environment for evaluating this particular metric.

Statistical interval set comparison

Our statistical interval comparison provides a structured way of describing the relationship between the values obtained under two different experimental conditions of an *individual metric*. Typically, though, experimental results examine more than one metric. Thus, we define a *statistical interval set* methodology with which to compare the performance under two different experimental conditions of a *set of metrics*. In particular, it is useful to know how *consistent* the relationships are from one metric to another in a set. For example, we might be able to say that all metrics in the set which measure “time” are Case (0), but all metrics in the set which measure “distance” are Case (12).

Rank ordering comparison

Another way we look at the relationships between physical and simulation is to examine the *rank ordering* in values of an individual metric obtained under a set of different experimental conditions. Take again the example of distance travelled. Supposed we want a robot to visit ten points in its environment, and we have five different ways of deciding the order in which the robot visits the points. Let’s call these `v1` through `v5`. We run experiments in both simulated and physical environments, and we compute the distance travelled for all five visiting methods. Then we sort the distance values, from shortest to longest, and obtain a rank-order for the corresponding visiting methods. We can do this for experiments conducted both in simulation and in the physical environment. If the rank-ordering is the same between the simulated and physical environments, then we can be confident that simulation is an effective method for comparing experimental conditions along the metric chosen. For example, in the sample human-robot experiment described above, if the distance travelled for the robots using `v1` is the shortest and the distance travelled for the robots using `v3` is the longest, in both physical and simulated environments, then we can be confident that the simulation environment produces results reliably similar ot the physical environment in order to be able to use the simulated environment for evaluating this particular metric across this set of behaviours.

Case Studies

We demonstrate the utility of our methodology with two case studies. The first case study involves a series of experiments that evaluate several different task allocation mechanisms for a multi-robot team. The second case study involves a series of experiments that evaluate two different mechanisms for interaction in a human-robot scenario. First we describe each case study, and then apply our four comparison methods to each: point-to-point comparison, statistical interval comparison, statistical interval set comparison and rank ordering comparison.

Case Study 1: Multi-robot task allocation

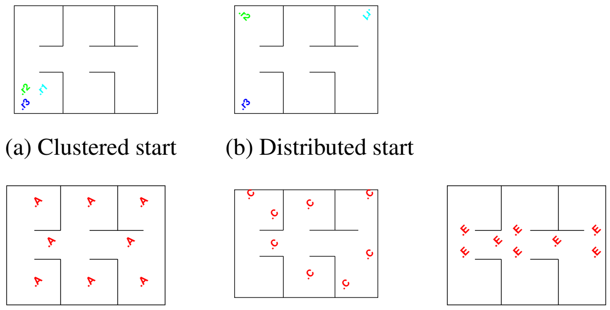
This case study involves a team of robots tasked to visit a number of *target points* in a constrained arena, organised such that one robot visits each point once. Our research in this case study concerns assessment of a number of different mechanisms by which tasks are allocated to robots. The results, with respect to allocation mechanisms, have been presented elsewhere (Özgelen et al., 2013; Schneider et al., 2014). Here, we are concerned with the comparison of results obtained in parallel *physical* and *simulated* settings.

Our experimental testbed employs a dual system architecture, based on *Player/Stage*[2] (Gerkey et al., 2003; Vaughan and Gerkey, 2007), in which both physical and simulated environments share common underlying system components. The details of our framework have been described elsewhere (Sklar et al., 2011).

There are two primary differences between the physical (Player) and simulated (Stage) instantiations of our framework: one is with respect to *localisation* and the other is with respect to robot *driving*.

Localisation refers to robots knowing where they are in their environment, in terms of a coordinate-based frame of reference. In the physical setup, this information is provided by a network of cameras, suspended above the arena, which track the robots and report their (x, y, θ) positions to all team members, through a central server process. In contrast, in the simulation setup, localisation is “perfect” because the simulator knows where all the robots are at all times. Thus, the physical environment is more *noisy* with respect to robots knowing where they are. *Driving* refers to robots knowing how to move, i.e., which motor(s) to turn on

[2] `http://playerstage.sourceforge.net/`


(a) Clustered start (b) Distributed start

(c) Target points A (d) Target points C (e) Target points E

Figure 3: Scenario definitions. (a) and (b) show starting locations. (c), (d), and (e) show target point set locations.

for how long. In the physical setup, a robot *controller* process communicates abstract motion commands (e.g., "forward") to a second *driver* process which converts the abstract motion command to platform-specific byte codes and transmits the codes to the physical robot. This abstraction of motion commands means that the only platform-specific element of the system is down at the driver level. In contrast, in the simulation setup, the same robot controller process sends the abstract motion commands to a robot driver in the simulator. Thus, the physical environment is again noisier than simulation and is also slower, because there is an extra level of communication (from the driver to the physical robots) that does not exist in simulation.

The experiments we conducted measured results in six different scenarios. All scenarios involved $n = 3$ robots and $m = 8$ target points. There were two sets of starting locations, one "distributed" and one "clustered", and three sets of target point locations (A, C and E). The starting locations and the target points can be seen in Figure 3. Experiments were conducted with each scenario using each of four different task allocation mechanisms. Here, we will refer to these generically as TAM1 through TAM4, because detailing the mechanisms is not the point of this paper (as above, details were reported elsewhere). Each combination was run 6 times in the physical environment and 30 times in the simulation environment.

$$\begin{aligned} start\ location &: \{\text{Clustered, Distributed}\}\times \\ targetpoint\ set &: \{\mathsf{A}, \mathsf{C}, \mathsf{E}\}\times \\ mechanism &: \{\mathsf{TAM1}, \mathsf{TAM2}, \mathsf{TAM3}, \mathsf{TAM4}\}\times \\ environment &: \begin{cases} \text{6 physical} & = \text{144 runs} \\ \text{30 simulation} & = \text{720 runs} \end{cases} \end{aligned}$$

For each run of each experiment, we recorded the following 6 metrics: (1) *deliberation time*, the total time required to allocate the target points; (2) *execution time*, the total time required to visit all the target points; (3) *distance travelled*, the total distance travelled by all robots to visit their assigned target points; (4) *idle time*, the total amount of time that robots were not executing a task, i.e., because they had no (more) target points to visit; (4) *delay time*, the amount of time robots spent avoiding collisions with others (explained below); and (6) *near collisions*, the number of times robots detected another's presence and stopped to negotiate right-of-way. Because each experiment involves many robots moving in a restricted area, they naturally get in each other's way. When robots are close enough to require evasive action, our system detects a "near collision," and the robots stop moving. Then the robot closest to its goal (current target point) is given the right-of-way. The other robot waits until its path is clear, and then continues on its way. The time that a robot was stopped for this reason is its *delay time*. In addition to measuring delay time, we counted the number of times that robots were delayed in this manner (*near collisions*).

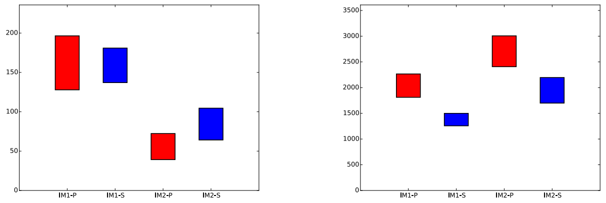
Case Study 2: Human-robot interaction

This case study involves a human-robot team collaboratively looking for "treasures" hidden in an environment that the robot can explore but the human cannot enter. In order for the team to complete the task—finding and correctly identifying all the treasures—the human and robot have to work together. There are tasks that only the robot can perform, such as wandering around in the environment and capturing images of what it "sees" there; and there are tasks that only the human can perform, such as identifying a particular treasure within an image (which is provided by the robot). Our research in this case study concerns assessment of an *argumentation-based dialogue* mechanism for facilitating human-robot collaboration. The results, with respect to interaction mechanism, have been presented elsewhere (Azhar et al., 2013; Sklar et al., 2013). Here, we are concerned with the comparison of results obtained in parallel physical and simulated settings.

The experimental setup is similar to that employed by Case Study 1. The robot explores the same physical environment as the multi-robot team. The physical robot is implemented using Player and the simulated version of the robot is implemented in Stage. The experiments we conducted measured not only the same 6 performance metrics described for Case Study 1, but also a number of metrics that assess the usability and impact of two interaction mechanisms. Here, we will refer to these generically as IM1 and IM2, because detailing the mechanisms is not the point of this paper (as above, details were reported elsewhere). We conducted a user study for Case Study 2, which involved 60 human subjects: 27 collaborated with a physical robot, and 33 collaborated with a simulated robot.

Results

Having presented each of our cases studies, here we apply our method for comparing the results of experiments to both



(a) Deliberation time

(b) Total distance

Figure 4: Deliberation time and distance travelled for the human-robot interaction case study. The left plot shows deliberation time, the right plot shows distance travelled. Each plots compares results for physical (red) and simulated (blue) across both forms of dialogue.

case studies in turn. We start with the human-robot interaction scenario since it is simpler.

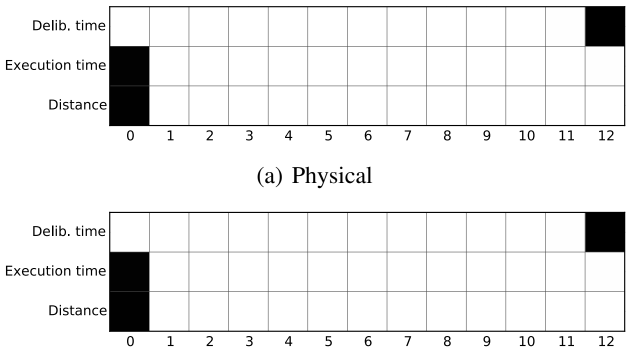
Human-robot interaction

Figure 4 shows the statistical interval method applied to the human-robot interaction scenario for two metrics, deliberation time and the total distance travelled. In this figure, we compare results obtained using physical robots against results obtained in simulation (red vs blue bars) to see if there are systematic differences. We see that for *total distance travelled*, the results obtained fall into interval relationship Case (12), where the results in simulation are consistently less than those on physical robots (and since we are using twice the standard deviation to construct the intervals, we can conclude that this difference is significant). For *deliberation time*, the results for IM1 fall into relationship Case (4) with the simulation results contained within the physical results (so any result in simulation is within what is found in practice) whereas the results for IM2 fall into relationship Case (2).

Figure 5 illustrates our "sets of intervals" analysis. Here for each metric, we compare experimental conditions—i.e., the two interaction mechanisms.

The key difference between this method and the method demonstrated above is that the statistical interval method (above) compares the red vs blue bars for each experimental condition, whereas here, the statistical interval set method compares the relationship between the two red bars vs the relationship between the two blue bars. Each cell in the heatmap in Figure 5 contains the number of pairs of results (i.e., two red bars for the physical plot at the top) which fall into each interval relationship case (numbered 0..12 across the x-axis of the plot). In this case, there is only one comparison (between the two conditions), so there is only one entry in each row in the heatmap.

But the idea of the heatmap is to make it easy to spot correspondences between physical and simulation results. Figure 5 shows that simulation and physical agree exactly on the comparative results for the two interaction mechanisms.



(a) Physical

(b) Simulation

Figure 5: Heatmap showing sets of interval relationships for the task allocation case study. Darker values represent higher counts.

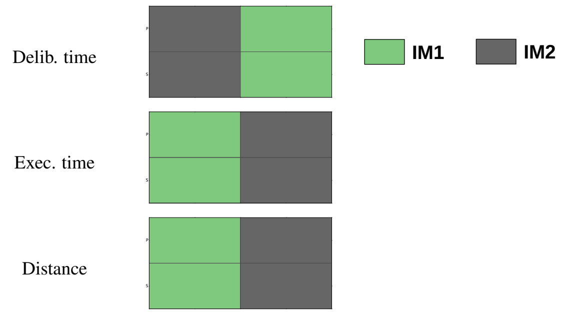


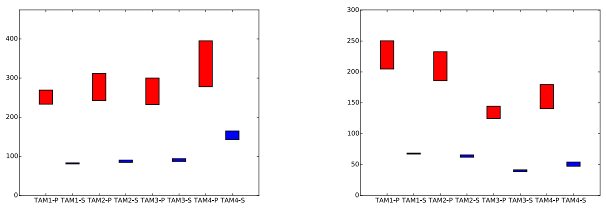
Figure 6: Rank order of metrics for the human-robot interaction case study. Values are ordered from left to right.

Figure 6 provides a way of looking at the rank order over metrics. These are rank-ordered, so that the top-valued condition is in the left column and the bottom-valued condition is in the right column. Since there are only two experimental conditions, this is not a complex plot; but as above, the idea is to make it easier to spot correspondences between physical (top row in each plot) and simulation (bottom row). Figure 6 shows perfect alignment in rank-ordering for the metrics illustrated.

Multi-robot task allocation

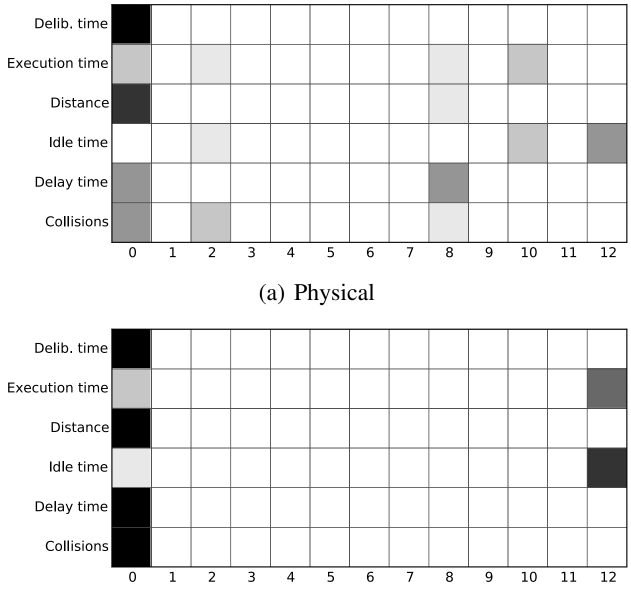
Figure 7 shows the comparison of the individual statistical intervals for one of the metrics and all the task allocations mechanisms. Here we use two standard deviations to define the intervals. In this particular case, it is easy to see that for this metric there is a consistent relationship between physical and simulated results across both start conditions. In all cases, the relationship between the intervals is Case (12), which allows us to say that the execution time for simulations is significantly less than that for physical robots.

Figure 8 shows the sets of intervals in heatmap form, a



(a) Clustered start (b) Distributed start

Figure 7: Execution time for the task allocation case study. The left plot shows the clustered start condition, the right plot shows the distributed start condition. Each plots compares results for physical (red) and simulated (blue) across all four task allocation mechanisms.



(a) Physical

(b) Simulation

Figure 8: Heatmap showing sets of interval relationships for the task allocation case study. Darker values represent higher counts.

more complex picture than in Figure 5. Figure 8 (a) summarises all the results on physical robots for two task allocation mechanisms, and (b) summarises all the results in simulation for the same two mechanisms. Each row in the heatmaps summarises all the results for a single metric across the different start configurations and the different sets of task points. For each experiment we generate the statistical intervals, establish which of the interval relationships that they fall into, and then count how many experiments stand in each of the thirteen interval relationships. Each cell then displays the relevant count, with darker cells reflecting a higher count.

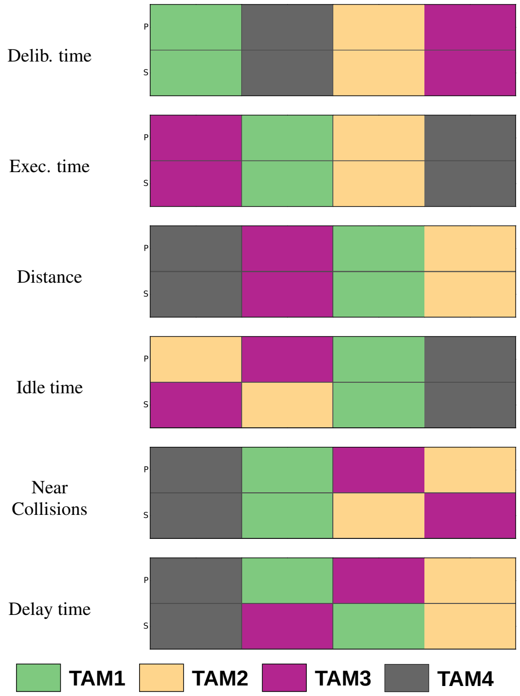


Figure 9: Rank order of metrics for the task allocation case study. Values are ordered from left to right.

What we are looking for here is similarity between physical and simulated experiments. For these two mechanisms, we can see strong agreement in terms of deliberation time: both physical and simulation have all comparisons falling into Case (0); good agreement on distance: all of simulation and most of physical fall into Case (0); and lesser agreement on delay time and collisions: simulated results all fall into Case (0), physical are split between Cases (0), (4) and (6); and idle time: all simulated are Case (3), while physical are split between Cases (3) and (7). These relationships allow us to identify when simulations will be good predictors of behaviour on real robots.

These relationships between mechanisms are shown even more clearly in Figure 9, which shows the application of the idea of rank-ordering to the results. In particular, this figure shows, for each metric, the relative performance of each mechanism across environments. Rankings are indicated by placement in the graphs, with the top-valued experimental condition (task allocation mechanism, in this case) on the left and the worst performer on the right. There are two rows for each metric: the top row indicates ranks for runs performed in the physical environment while the bottom indicates those for thesimulation environment. These results show quite clear agreement of rank orders in physical and simulation environments. Deliberation time, execution time, and distance travelled show exact agreement. The remaining metrics–idle time, delay time, and number of near collisions–are "misaligned" by at most one rank order.

Summary and Future Work

The interval relationships developed in this work, and the corresponding graphical representations give an immediate and, at the same time, nuanced indication of the way physical and simulation environments relate to one another. We have presented three different ways of comparing results obtained in parallel physical and simulation environments, and we have demonstrated these methods of comparison in two different case studies.

There are a number of open issues that have emerged as result of our preliminary work presented here. First, it is a logical question to ask what the methods presented here provide that traditional *t*-tests from statistics do not tell us. Second, another natural question to ask is how the methods presented here hold up in the face of significant stochasticity in results (e.g., where 30 runs does not offer convergence or demonstrate normal distributions), as well as handling of outliers. Finally, the methodology presented examines scalar values that vary from run to run, but do not trend over time. In the case of domains that involve learning, such as any attempt to model human behaviour, certain statistics will improve as the human learns. This kind of situation, where the metric being compared is a function rather than a scalar value, will require different treatment. These questions will be investigated in future work.

Acknowledgements

Thanks to Gal Kaminka who first suggested Allen's intervals in connection with our work. Part of this work was completed while some of the authors were with Brooklyn College and Hunter College of the City University of New York (CUNY). We gratefully acknowledge the work of our CUNY colleagues including Ofear Balas and Michael Squitieri who contributed to the software development and running of some of the experiments contained this paper. This work was partially supported by grant #IIS-11-16843 from the US National Science Foundation (NSF).

References

Allen, J. F. (1981). An interval-based representation of temporal knowledge. In *7th International Joint Conference on Artificial Intelligence*, pages 221–226.

Allen, J. F. (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11).

Azhar, M. Q., Schneider, E., Salvit, J., Wall, H., and Sklar, E. I. (2013). Evaluation of an argumentation-based dialogue system for human-robot collaboration. In *Workshop on Autonomous Robots and Multirobot Systems*, St Paul, MN, USA.

Brooks, R. A. (1992). Artificial life and real robots. In *1st European Conference on Artificial Life*, pages 3–10. MIT Press.

Farchy, A., Barrett, S., MacAlpine, P., and Stone, P. (2013). Humanoid robots learning to walk faster: From the real world to simulation and back. In *12th International Conference on Autonomous Agents and Multiagent Systems*, pages 39–46.

Fox, D., Burgard, W., and Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33.

Gerkey, B., Vaughan, R. T., and Howard, A. (2003). The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. In *11th International Conference on Advanced Robotics*.

Jakobi, N., Husbands, P., and Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in Artificial Life*, pages 704–720. Springer.

Koos, S., Mouret, J.-B., and Doncieux, S. (2010). Crossing the reality gap in evolutionary robotics by promoting transferable controllers. In *12th Annual Conference on Genetic and Evolutionary Computation*, pages 119–126. ACM.

Koza, J. R. (1991). Evolving emergent wall following robotic behavior using the genetic programming paradigm. In *First European Conference on Artificial Life*, pages 110–119, Cambridge, MA. MIT Press.

Marques, H. G. and Holland, O. (2009). Architectures for functional imagination. *Neurocomputing*, 72(4-6):743–759.

Mataríc, M. J. (1990). A distributed model for mobile robot environment-learning and navigation. Technical report, MIT Artificial Intelligence Laboratory.

Özgelen, A. T., Schneider, E., Sklar, E. I., Costantino, M., Epstein, S. L., and Parsons, S. (2013). A first step toward testing multiagent coordination mechanisms on multi-robot teams. In *Proceedings of the Workshop on Autonomous Robots and Multirobot Systems*.

Schneider, E., Balas, O., Özgelen, A. T., Sklar, E. I., and Parsons, S. (2014). An Empirical Evaluation of Auction-based Task Allocation in Multi-Robot Teams (Extended Abstract). In *13th International Conference on Autonomous Agents and Multiagent Systems*, Paris, France.

Sklar, E., Ozgelen, A. T., Munoz, J. P., Gonzalez, J., Manashirov, M., Epstein, S. L., and Parsons, S. (2011). Designing the HRTeam framework: Lessons learned from a rough-and-ready human/multi-robot team. In *Workshop on Autonomous Robots and Multirobot Systems*, Taipei, Taiwan.

Sklar, E., Ozgelen, A. T., Schneider, E., Costantino, M., Munoz, J. P., Epstein, S. L., and Parsons, S. (2012). On transfer from multiagent to multi-robot systems. In *Workshop on Autonomous Robots and Multirobot Systems*, Valencia, Spain.

Sklar, E. I., Azhar, M. Q., Parsons, S., and Flyr, T. (2013). A Case for Argumentation to Enable Human-Robot Collaboration (Extended Abstract). In *12th International Conference on Autonomous Agents and Multiagent Systems*, St Paul, MN, USA.

Vaughan, R. and Zuluaga, M. (2006). Use your illusion: Sensorimotor self-simulation allows complex agents to plan with incomplete self-knowledge. In *From Animals to Animats 9*, volume 4095 of *Lecture Notes in Computer Science*, pages 298–309.

Vaughan, R. T. and Gerkey, B. (2007). Really Reusable Robot Code and the Player/Stage Project. In Brugali, D., editor, *Software Engineering for Experimental Robotics*. Springer.

Achieving Closure in Enzymes in Artificial Chemistries

Matteo Monti[1,3], Tim Hutton[2] and Piet Hut[3,4]

[1]University of Bologna, 40126 Bologna, Italy
[2]Independent researcher
[3]Institute for Advanced Study, Princeton, NJ08540, USA
[4]Earth-Life Science Institute, Tokyo Institute of Technology, Meguro, Tokyo 152-8551, Japan
monti@ias.edu

Abstract

Previous work has shown simulations of self-reproducing cells within the framework of a two-dimensional artificial chemistry. This earlier work used either a set of fixed rules to produce specific behaviors, or atomic enzymes governed by an extended set of meta rules. The difficulty with such enzymes is that their method of operation is outside of the system and thus cannot itself be reprogrammed by enzymes, unless another meta-level of rules is added - the system lacks closure.

Here we generalize the rules by introducing molecular enzymes that have catalytic functions capable of being extended to the very same components of enzymes themselves, without changing the underlying system rules. This allows for a pool of enzymes to reproduce themselves while remaining fully embedded in the arena of competition. Together with the limited availability of components this permits each organism to become a resource for the others, thus enabling rich interactions between the organisms not only via their shared environment (e.g. waste products) but also directly (e.g. attacking each other). With stochastic mutations we suggest that the ingredients are thus in place for open-ended evolution.

How Intrinsic Motivation can Speed Up Language Emergence

Miquel Cornudella, Paul Van Eecke and Remi van Trijp

Sony Computer Science Laboratory, Paris
6 rue Amyot, 75005, Paris, France
cornudella@csl.sony.fr — vaneecke@csl.sony.fr — remi@csl.sony.fr

Abstract

Natural languages enable humans to engage in highly complex social and conversational interactions with each other. Alife approaches to the origins and emergence of language typically manage this complexity by carefully staging the learning paths that embodied artificial agents need to follow in order to bootstrap their own communication system from scratch. This paper investigates how these scaffolds introduced by the experimenter can be removed by allowing agents to autonomously set their own challenges when they are driven by intrinsic motivation and have the capacity to self-assess their own skills at achieving their communicative goals. The results suggest that intrinsic motivation not only allows agents to spontaneously develop their own learning paths, but also that they are able to make faster transitions from one learning phase to the next.

Introduction

Natural languages enable humans to engage in highly complex social and conversational interactions with each other. Their intricacies have seduced a host of Alife researchers who employ multi-agent experiments for modeling the origins and emergence of natural language-like communication systems (e.g. Kirby, 1999; Smith et al., 2003; Steels, 2012; Wagner et al., 2003). These experiments typically manage the complexity of language by carefully scaffolding the developmental stages which embodied interacting agents have to go through in order to bootstrap their own communication system. In real life however, such scaffolds obviously do not exist, so it is important to investigate how agents can autonomously set their own goals in order to manage the world's complexity.

Recently, there has been an important movement that looks at the role of *intrinsic motivation* and *skill self-assessment* for this purpose. The role of motivation has already been extensively studied by psychologists (Hull, 1943; Skinner, 1953; White, 1959; Graham, 1996) who have a.o. observed that children are remarkably fast at developing new competences. We define "motivation" as "to be moved to do something" (Ryan and Deci, 2000), and further subdivide the notion into *extrinsic motivation*, where external forces influence motivated behavior, and *intrinsic motivation*, which consists of performing activities because they are experienced as inherently enjoyable or interesting.

Roboticists and AI researchers have become increasingly interested in operationalizing these mechanisms in order to enable embodied agents to become robust learners in open-ended environments. Many promising results have already been achieved, particularly in the domain of behaviors, by looking at interest (Merrick and Maher, 2009), error reduction (Andry et al., 2001; Roy and McCallum, 2001), reinforcement (Bonarini et al., 2006), novelty (Huang and Weng, 2002; Barto et al., 2004), prediction (Marshall et al., 2004) and curiosity (Kaplan and Oudeyer, 2007; Oudeyer et al., 2007). Interested readers are referred to Baldassarre and Mirolli (2013) for an extensive overview.

This paper aims to contribute to this field by examining the role of intrinsic motivation in language emergence. More specifically, it introduces an agent-based experiment in which a multi-agent population of artificial agents are able to self-organize a vocabulary and to extend that vocabulary to a primitive syntactic language that allows the agents to use multiword utterances for referring to objects in their environment. Besides the required mechanisms for adopting and inventing words and patterns, the agents are also equipped with a motivation drive that allows them to set their own communicative goals, and with the capacity to assess their own skill level at solving particular communicative tasks. The motivational drive is inspired by Flow theory (Csikszentmihalyi, 1990) and builds further on earlier Alife experiments that have proposed concrete operationalizations of Flow (Steels, 2004).

Flow Theory

The approach used in this work for modeling intrinsic motivation in a population of artificial agents is based on the Flow theory proposed by Csikszentmihalyi (1990), which aims to explain why people get involved in complex activities that do not imply a clear external reward. Csikszentmihalyi observed that humans do so because these activities induce a strong feeling of enjoyment in their participants.

He called these activities *autotelic*, meaning that it is the individual him/herself (*auto*) who is the motivational driving force (*telos*).

Csikszentmihalyi introduces two core concepts to the theory of flow: *challenge*, which refers to how difficult a particular task or activity is, and *skill*, which refers to the abilities of an individual. As shown in Figure 1, any activity can be conceptualized as an interaction between challenge (x-axis) and skill (y-axis). If an activity is too challenging with respect to an individual's skill level, it induces a feeling of anxiety. Conversely, if a participant has a high skill level, but the challenge is too low, the experience becomes boring. According Csikszentmihalyi, participants may therefore try to strike a balance between skill and challenge, in which they may experience strong enjoyment. Csikszentmihalyi calls such an experience *flow*.

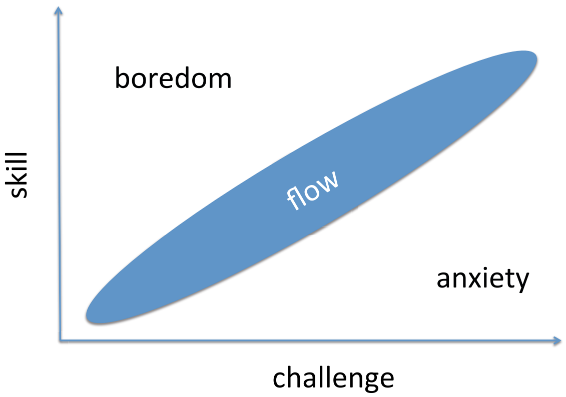


Figure 1: Csikszentmihalyi (1990) argues that individuals may enter a state of *flow* (i.e. intrinsic enjoyment) when they carefully balance their skill levels against the challenges that they select when performing particular activities. Distortion of this balance may lead to anxiety (i.e. the challenge is too big) or boredom (i.e. the challenge is too easy).

By the dynamics of the flow theory, the flow state is in continuous change. In order to progress in developing his/her abilities, an individual involved in an autotelic activity must keep him/herself in the flow state. To do so, s/he must be able to decrease his/her challenge when his/her skills are too low (to avoid anxiety) and increase his/her challenge when his/her skills are too high, to avoid boredom.

The Autotelic Principle

Steels (2004) presented a concrete operationalization of Flow theory through the *autotelic principle*, which we adopt in this paper. The autotelic principle allows agents to set their own challenges by monitoring their own performance for determining their *emotional state* given a particular skill/challenge payoff. Depending on their emotional state, agents will decide to move to higher challenges (in case of boredom) or fall back to easier challenges (in case of anxiety), or they continue to develop skills for coping with the current challenge (when they achieve a balance between skill and challenge). The Autotelic Principle has been applied in experiments in the domain of behavioral robotics (Steels, 2005) and, particularly relevant for this paper, language (Steels and Wellens, 2007).

More specifically, the architecture of the autotelic principle includes the following parts:

- Parameterized tasks: Embodied agents who interact with each other and their environment need to perform many tasks, such as perceiving the world, conceptualizing and interpreting meanings, and producing or comprehending utterances. Each of these tasks must be parameterized to reflect different challenge levels. For example, one parameter for the task of describing an object could be the number of words that an agent can employ. Formally, we associate each task t_i with a parameter vector $< p_{i,1}, ..., p_{i,n} >$. The set of all parameters for all tasks forms a multidimensional parameter space P. At any point in time, agents *self-regulate* their behavior by taking a particular configuration of these parameters $p(s,t) \, in \, P$.

- Self-monitoring: Self-regulation of behavior is operationalized as a search process in the multi-dimensional parameter space P to maintain acceptable performance. In order to calculate the most optimal parameter configuration $p(s,t) \ in \ P$, the agents need to be capable of monitoring their performance for all tasks. In our paper, we implement monitors for each task that yield a value in the range [0, 1] with 1 being the optimal performance for that task. The performance of an agent at a given time is measured by simply averaging the sum of the performance of all monitors actively used by the agent. This measure is used to calculate the *confidence* an agent has in its skill level for achieving a particular task. After each interaction, confidence is updated taking into account the performance of an agent and the success or failure in the interaction.

- Shake-up phase: The shake-up phase takes place once an agent has acquired sufficient experience with a given parameter setting $p(s,t) \ in \ P$. Sufficient experience means that the confidence level of the agent does not show significant fluctuations any more during specific observation windows (i.e. the confidence value in the n latest interactions). The goal of the shake-up phase is to adjust the challenge parameters when this situation occurs. Two scenarios are possible: (a) confidence is too low, so the agents are stuck in a state of Anxiety and will fall back to an easier challenge; and (b) confidence is consistently high, so the agents are stuck in a state of Boredom and so will try to increase the challenge level (e.g. by selecting more ambitious communicative goals).

Experiment

Our experiment aims to show that intrinsic motivation allows a population of agents to develop learning paths at their own pace (that is, without predefined scaffolds imposed by the experimenter) and that it may speed up the emergence of a community language. This shared language is developed by the agents through recurrent communicative interactions, which take place following a well-defined script, also called *language game* (Steels, 2012).

The language game

The specific game that agents play in our experiment is a *multiword guessing game* and proceeds as follows. Each game is played by two agents that were randomly selected from a population N with population size $N = 10$. One of the agents is assigned the role of speaker, the other agent the role of listener.

The world of the agents consists of ten different scenes, with each scene consisting of two different objects. Objects are described using feature-value pairs, which we have given human understandable labels for mnemonic reasons: prototype (e.g table, chair, cup), shape (e.g. pentagonal, square, round) and color (e.g. blue, green, purple). For each scene, speaker and listener build up the same *situation model*, which is assumed to be grounded in the world through perception and which is described using a first order predicate calculus. Example (1) shows a possible situation model s_k.

(1) $\exists obj1, obj2 :$ { *prototype(obj1, table), shape(obj1,round), color(obj1, green), prototype(obj2, table), shape(obj2, square), color(obj2, brown)* }

At each interaction, one specific scene is randomly selected from the world, for both speaker and listener. Additionally, the speaker chooses a *topic*, which he should express to the listener. The topic is either one of the objects or both objects. The task of the speaker is to produce an utterance (transmitted as text) conveying information about the topic in such a way that the listener is able to identify the topic in the scene. In order to achieve this, the speaker can refer to the prototype, shape or color of one or more objects. The listener tries to comprehend the utterance and makes hypotheses about what the topic is. If he has only one hypothesis, he points to the hypothesized topic and gets feedback from the speaker about whether this was indeed the topic. If his pointing turns out to be correct, the game is successful. If not, the game fails and the speaker points at his topic as a means of providing corrective feedback. If the listener has multiple hypotheses, he communicates to the speaker that he could not identify the topic. In this case, the game also fails and the speaker points to the topic. After each interaction, both speaker and listener go through an evaluation phase in which they evaluate their performance and, if possible, learn from the feedback provided by the other agent.

Implementation

We will now have a closer look at how some crucial parts of the language game are implemented, in particular the invention of new words, the adoption of words used by an other agent and the evaluation of produced and comprehended utterances.

The agents start the experiment with given *conceptualization* and *interpretation* mechanisms. These mechanisms allow the agents to map between objects in their world model and meanings that refer to these objects. A red box in their world for example, can be mapped to the meaning `box(?x), red(?x)`. Although the agents start with fully operational mechanisms for mapping between meanings and objects in their world model, they start without any mapping from a meaning to an utterance (production) or from an utterance to its meaning (comprehension). It is this mapping that emerges through the recurrent interactions.

The agents have three mechanisms which allow them to develop, adopt and agree upon a shared language: *diagnostics*, *repairs* and *alignment*. Diagnostics allow agents to identify problems during production or comprehension. Repairs are strategies that agents use to solve diagnosed problems. Alignment keeps track of the success rate of form-meaning mappings in every agent and guides the choice of which mappings the agent uses.

Diagnostics and Repairs Agents can diagnose *uncovered meanings*, *uncovered strings* and problems of *word order* and *reference*. Uncovered meanings can be diagnosed by the speaker when he tries to express a meaning predicate that is not covered by his construction inventory. It is solved with a repair that creates a new word for the uncovered meaning. Uncovered strings correspond to the analogous problem for the listener. If the listener identifies a word for which he does not have a form-meaning mapping in his construction inventory, he can use the feedback of the speaker to infer the meaning of this form and create the corresponding form-meaning mapping. Note that this is only possible in interactions where the combination of the feedback of the speaker and the knowledge of the listener allow to unambiguously infer the meaning of this word. For example, if a listener fails to comprehend a three word utterance because he does not know any of the words, he will not be able to infer which meaning each word expresses and does not learn any new mappings.

The word order and reference problems can occur when agents refer to more than one feature of an object. Agents start without mechanisms specifying that two words refer to different features of the same object or of two different objects. These problems can be solved by creating grammatical constructions that introduce ordering constraints.

Alignment Whereas diagnostics and repairs provide the agents with mechanisms to invent new words and to adopt words used by the other agents, alignment steers the preference of agents to use certain form-meaning mappings. Every form-meaning mapping has a score, a number between 0 and 1. If the construction inventory of an agent contains multiple form-meaning mappings that can be used to express a certain meaning (*competitors*), he will choose the one with the highest score. For a new form-meaning mapping, which is either invented or adopted, the score is initialised at 0.5 and after each interaction, the agents update the score of the constructions in their lexicon. As a side-effect of the alignment, the construction inventories of the agents in the population converge to a minimal, shared set of constructions.

The alignment used in our experiment updates the scores according to the dynamics of *lateral inhibition* (Vylder and Tuyls, 2006). The alignment algorithm is shown in algorithm 1. If speaker and listener reached communicative success, they both increase the score of the constructions they applied by 0.1 and decrease the score of the competing constructions by 0.1. For the speaker, competing constructions are those that express the same meaning as the constructions that he used (*meaning competitors*), whereas for the listener, they are those constructions that express the same form (*form competitors*). In the case that the communication was not successful, the alignment is different for speaker and listener. The speaker only aligns if he produced a one-word utterance, in which case he decreases the score of the applied construction by 0.1. The listener only aligns when the speaker's topic was among his hypotheses. Then, the listener increases the score of the applied constructions by 0.1 and decreases the score of its competitors by 0.1. In other cases, the speaker and listener don't align, as they cannot be sure which constructions have to be rewarded and which ones punished. Note that the upper bound for the score of a construction is 1 and that constructions which reach a score of 0 are deleted from the agents construction inventory.

```
// For the speaker
if Communicative Success then
    score of applied constructions + 0.1;
    score of meaning competitors - 0.1;
else
    if one-word utterance then
        score of the applied construction - 0.1;
    end
end
// For the listener
if Communicative Success then
    score of applied constructions + 0.1;
    score of form competitors - 0.1;
else
    if topic among hypotheses then
        score of the applied constructions + 0.1;
        score of the form competitors - 0.1;
    end
end
```

Algorithm 1: The alignment algorithm

The Baseline Experiment

In order to show that a model of intrinsic motivation can indeed speed up language development, we first need to establish a baseline. The aim of the baseline experiment is to reveal how many interactions it takes before all agents in the population are able to consistently fulfill a given communicative task. The baseline experiment uses the guessing game script described in the previous section. The speaker always refers to one or two objects in the scene, and minimally expresses the *prototype* of the object(s). Apart from the prototype, the speaker can refer to one or more properties of the objects he wants to describe. The maximum number of properties that the speaker can express is specified by the *learning tasks*:

- Learning task 1: Agents can refer to up to two objects only referring to their prototypes.
- Learning task 2: Agents can refer to up to two objects only referring to their prototypes and one property.
- Learning task 3: Agents can refer to up to two objects only referring to their prototypes and two properties.
- Learning task 4: Agents can refer to up to two objects only referring to their prototypes and three properties.
- Learning task 5: Agents can refer to up to two objects only referring to their prototypes and four properties.

Every learning task also includes the previous learning tasks. For example, in learning task 3, agents can describe two properties, one property or no properties at all. Note that the learning tasks specify the total number of prototypes that the speaker can express, not the number of properties per object. Moreover, the learning tasks do not specify whether the properties are associated with one object or with two objects. In learning task 3 for example, the speaker can refer to the prototype and two properties of an object, to the prototypes of two objects and one property of each object, or to the prototypes of two objects and two properties of one of them.

Each learning task has been run 10 times in a population of 10 agents. The average results over these runs are shown in figure 2. The x-axis represents the number of communicative interactions (i.e. the number of times the guessing game is played) and the y-axis represents the communicative success. The communicative success has a value of either 1 or 0 after every interaction, but has for clarity reasons been smoothed by a sliding window of 100 interactions. The experimental results show that for learning tasks in which the speaker refers to a smaller number of properties, a shared language emerges faster than for learning tasks in which the speaker refers to more properties. On the other hand, the communicative success stabilizes at a lower percentage for lower tasks than for the higher tasks.

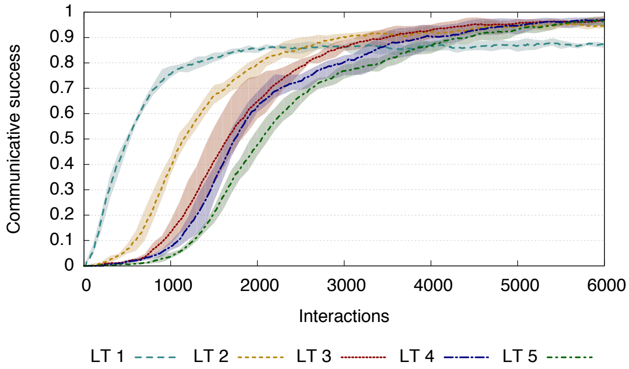


Figure 2: Communicative success for the different learning tasks in a population of 10 agents. As expected, agents are faster at learning tasks that have a lower challenge. However, since the utterance types associated with these easier challenges are often insufficient for properly discriminating the topic in a scene, communicative success stagnates at a lower rate.

The fact that the lower learning tasks are learned faster can be explained by the fact that the shorter the utterances are, the easier it is for the agents to learn and agree on form-meaning mappings. For longer utterances, it is often not possible to infer the meaning of the different words and adopt the relevant mappings. The fact that the communicative success stabilizes on a lower percentage for the lower learning tasks than for the higher ones is due to the ambiguity in the topic descriptions. Referring to a smaller number of properties decreases the discriminative power of the utterance. Referring to prototypes only for example, does not suffice to discriminate a blue, triangular table from a red, round one. Indeed, when monitoring the ambiguity of the scenes for the different learning tasks, ambiguity turned out to explain exactly the percentage of games in which communicative success could not be achieved. For learning task 1 for example, about 13,5% of the topic descriptions were ambiguous, whereas for learning task 5, this was only 1.7%. In fact, when agents refer to four properties, there is no ambiguity in the scene at all. But as every learning task also includes the lower learning tasks, some of the utterances include fewer properties, leading indeed to small percentages of ambiguity.

Integration of intrinsic motivation

The number of interactions it takes for a population to master a learning task indicates how difficult the task is for the agents. In our next experiment, we integrate the model of intrinsic motivation described above, in which the learning tasks from the baseline experiment serve as *challenges*. Agents develop a lexicon and primitive grammar to refer to the objects in their environment, starting with the easier challenges and gradually shifting to the more difficult ones. The agents autonomously decide to move to a more difficult challenge when they master an easier one (and feel Bored) or to go back to an easier task if they are not successful enough in the new task (and feel Anxious).

At the beginning of the experiment, all agents start with the challenge corresponding to learning task 1, as the baseline experiment showed that this is the easiest one to learn. After each interaction, agents calculate their *confidence* in the task. The confidence score of the agents is calculated based on the result of the interaction (whether the communication was successful or not) and their performance, i.e. whether the agents could successfully produce, comprehend or learn. The confidence score of an agent for a challenge ranges from 0.0 to 1.0 and corresponds to his emotional state. When an agent reaches a score of 1.0, he feels Bored and moves to a more difficult challenge. Conversely, when the confidence score of an agent reaches 0.0, the agent feels Anxious and reacts by moving back to a lower challenge.

The results of 10 experimental runs for a population of 10 agents equipped with the autotelic principle are shown in figure 3. The x-axis represents the number of communicative interactions in the population. The left y-axis indicates the rate of communicative success (again smoothed with a sliding window of 100 interactions), and the right y-axis reflects the number of challenges that the agents master (measured by calculating the average confidence score of all agents in the population over all challenges).

We can observe that the agents develop a shared language for referring to prototypes (the initial challenge) in approximately 1200 interactions. At that moment, they master the first challenge, experience Boredom and move on to the second challenge. Then, the communicative success in the population drops. This can be explained by the fact that at this point, the agents don't only refer to prototypes any more, but also start to refer to other properties of the objects, such as shape and color. Moreover, they need to develop grammatical means for managing reference issues in multiword utterances. By the time that the agents master the second challenge, their vocabulary for prototypes and properties is

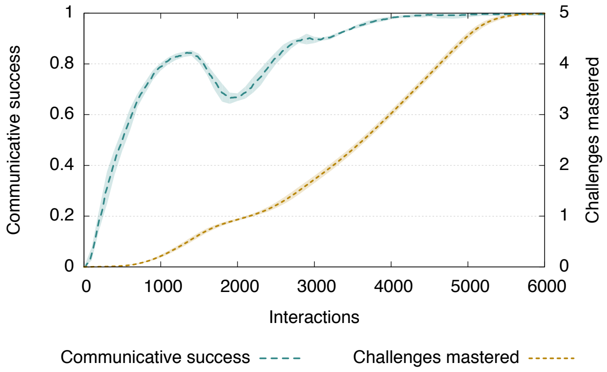


Figure 3: This graph shows communicative success (left y-axis) and the number of challenges mastered by the agents (right y-axis) in a population of 10 agents equipped with the autotelic principle. As can be seen, agents rapidly reach maximum success, with a small decline between language games 1200 and 2000, when they move up to more challenging communicative goals. After roughly 6000 interactions (an average of 1200 per individual agent), all agents in the population have reached the same skill level.

fully developed. This drastically speeds up the learning of the remaining challenges, as they now only need to further develop grammatical constructions to refer to more properties of an object.

As the language development process of each agent is shaped by the specific interactions that he participated in, some agents master a challenge more rapidly and move to a higher challenge faster than other agents. When speaking to agents that are still in a lower challenge, the communication is very likely to fail, as the listener will not be able to understand the more complex utterance. This phenomenon slightly decreases the confidence score of the agents in the lower challenge in phases where the population gradually shifts from one challenge to another. This makes the rate at which the confidence score of the agents increases slow down. In figure 3, this effect is best visible at the transition from challenge 1 to challenge 2, as this transition needs the most learning time.

Discussion

The results of our experiments show that a population of artificial agents is indeed able to self-organize a vocabulary and primitive syntactic language that allows the agents to communicate about their environment using multi-word utterances. In the experiment including the autotelic principle, the agents managed to evaluate their performance and autonomously adjust their challenge level to stay in a state of flow. Although each agent individually decided when to move up to a next challenge, the population as a whole reached the maximum skill level.

The development of language using the autotelic principle prevents agents from using too complex descriptions (and so using too many words or grammatical constructions that have not yet spread in the population) before a considerable part of the population can understand them. This avoids many interactions in which the utterance produced by the speaker is too difficult for the listener to learn from. This facilitates the learning of the basic skills, which is expected to result in a faster development of a shared language for complex learning tasks.

Figure 4 and 5 show the number of interactions it takes on average for an agent to reach maximal confidence in the different learning tasks, with and without making use of the autotelic principle respectively. The x-axis represents the number of interactions per agent and the y-axis the average confidence score of the agent in a learning task.

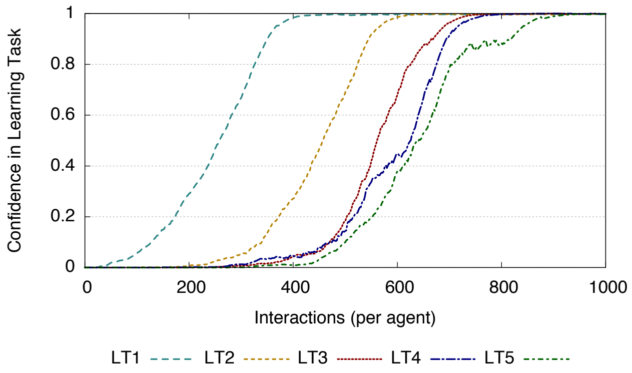


Figure 4: This graph shows how many interactions an agent needs on average for reaching maximal confidence in a particular task *without* the autotelic principle. This baseline result shows that each learning task indeed poses greater challenges on learning agents, with LT5 taking almost 1000 interactions before it can be mastered.

We can observe that when the agents do not make use of the autotelic principle, each learning task poses indeed a greater challenge to the agents than the previous one, with learning task 5 taking almost 1000 interactions before it can be mastered. When the agents do make use of the autotelic principle however, we see that, having acquired the lower challenges first, they reach maximal confidence for the higher challenges much faster.

The experiments clearly show that the use of challenges set by the individual agents speeds up the learning of more challenging tasks. The expectation is that in the experiment using the autotelic principle, the cumulative number of interactions needed for reaching maximal confidence in all tasks would also be lower than for reaching maximal confidence

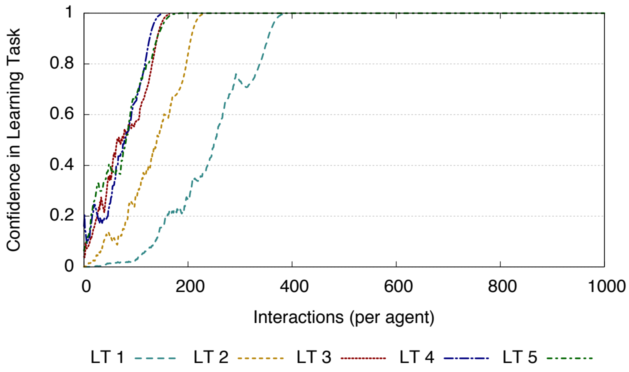


Figure 5: This graph shows how many interactions it takes on average for an agent to become confident in a learning task *with* the autotelic principle. The results demonstrate that agents become better and faster learners: each time they move to a more complex challenge, they are able to reach maximum confidence in less time than needed for the previous challenge.

in the most difficult task without the autotelic principle. The current experiment does not include enough challenges of differing difficulties to draw conclusions about this global effect, which will therefore be studied in further research.

Conclusions

In this paper, we have studied the impact of intrinsic motivation on language development in a population of artificial agents. We have equipped the agents with a model of motivation based on the autotelic principle, inspired by the theory of Flow. This mechanism allows the agents to develop their skills in a progressive fashion, regulating the complexity of their challenges by monitoring their own performance. In our experiment, the population of agents developed a shared language through recurrent communicative interactions. We defined five communicative tasks of different complexity that the agents had to fulfill. The results show that the population of agents managed indeed to progressively shift from easier to more difficult tasks, as a consequence of every individual agent striving to stay in a state of flow. Moreover, agents needed progressively less interactions to reach confidence in more difficult tasks, as they did not spend any interactions on utterances that are too complex to learn from.

Acknowledgements

This research was conducted at and funded by the Sony Computer Science Laboratory Paris. Additional funding was provided by the ANRT, CIFRE grant agreement no. 2013/0730 (Cornudella) and the Marie Curie Initial Training Network Essence, grant agreement no. 607062 (Van Eecke). The authors wish to thank their colleagues for their feedback and support, particularly Luc Steels and Emília Garcia Casademont. The authors also would like to thank Thierry Poibeau for his support.

References

Andry, P., Gaussier, P., Moga, S., Banquet, J.-P., and Nadel, J. (2001). Learning and communication via imitation: An autonomous robot perspective. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(5):431–442.

Baldassarre, G. and Mirolli, M. (2013). *Intrinsically motivated learning in natural and artificial systems*. Springer.

Barto, A. G., Singh, S., and Chentanez, N. (2004). Intrinsically motivated learning of hierarchical collections of skills. In *Proc. 3rd Int. Conf. Development Learn*, pages 112–119.

Bonarini, A., Lazaric, A., Restelli, M., and Vitali, P. (2006). Self-development framework for reinforcement learning agents. In *Proceedings of the Fifth International Conference on Development and Learning*, volume 178, pages 355–362. Citeseer.

Csikszentmihalyi, M. (1990). *Flow: The psychology of optimal experience*. Harper and Row, New York.

Graham, S. (1996). Theories and principles of motivation. *Handbook of educational psychology*, 4:63–84.

Huang, X. and Weng, J. (2002). Novelty and reinforcement learning in the value system of developmental robots. In *Lund University Cognitive Studies*, pages 47–55.

Hull, C. L. (1943). *Principles of behavior: an introduction to behavior theory.* Appleton-Century.

Kaplan, F. and Oudeyer, P.-Y. (2007). The progress-drive hypothesis: an interpretation of early imitation. *Models and mechanisms of imitation and social learning: Behavioural, social and communication dimensions*, pages 361–377.

Kirby, S. (1999). Syntax out of learning: The cultural evolution of structured communication in a population of induction algorithms. In Floreano, D., Nicoud, J., and Mondada, F., editors, *Advances in Artificial Life: Proceedings of the 5th European Conference on Artificial Life*, pages 694–703, Berlin. Springer.

Marshall, J., Blank, D., and Meeden, L. (2004). An emergent framework for self-motivation in developmental robotics. In *Proceedings of the 3rd international conference on development and learning (ICDL 2004), Salk Institute, San Diego*, volume 10.

Merrick, K. and Maher, M. L. (2009). Motivated learning from interesting events: adaptive, multitask learning agents for complex environments. *Adaptive Behavior*, 17(1):7–27.

Oudeyer, P.-Y., Kaplan, F., and Hafner, V. V. (2007). Intrinsic motivation systems for autonomous mental development. *Evolutionary Computation, IEEE Transactions on*, 11(2):265–286.

Roy, N. and McCallum, A. (2001). Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*.

Ryan, R. M. and Deci, E. L. (2000). Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary educational psychology*, 25(1):54–67.

Skinner, B. F. (1953). *Science and human behavior*. Simon and Schuster.

Smith, K., Kirby, S., and Brighton, H. (2003). Iterated learning: A framework for the emergence of language. *Artificial Life*, 9(4):371–386.

Steels, L. (2004). The autotelic principle. In Fumiya, I., Pfeifer, R., Steels, L., and Kunyoshi, K., editors, *Embodied Artificial Intelligence*, volume 3139 of *Lecture Notes in AI*, pages 231–242. Springer Verlag, Berlin.

Steels, L. (2005). The emergence and evolution of linguistic structure: from lexical to grammatical communication systems. *Connection Science*, 17(3-4):213–230.

Steels, L. (2012). *Experiments in Cultural Language Evolution*. Advances in interaction studies. John Benjamins Publishing Company.

Steels, L. and Wellens, P. (2007). Scaffolding language emergence using the autotelic principle. In *Artificial Life, 2007. ALIFE'07. IEEE Symposium on*, pages 325–332. IEEE.

Vylder, B. D. and Tuyls, K. (2006). How to reach linguistic consensus: A proof of convergence for the naming game. *Journal of Theoretical Biology*, 242(4):818 – 831.

Wagner, K., Reggia, J. A., Uriagereka, J., and Wilkinson, G. S. (2003). Progress in the simulation of emergent communication and language. *Adaptive Behavior*, 11(1):37.

White, R. W. (1959). Motivation reconsidered: the concept of competence. *Psychological review*, 66(5):297.

Coordination of collective behaviours in spatially separated agents

Rob Mills[1], Payam Zahadat[2], Fernando Silva[1],
Damjan Mlikic[3], Pedro Mariano[1], Thomas Schmickl[2] and Luís Correia[1]

[1] BioISI – Biosystems & Integrative Sciences Institute, Faculty of Sciences, University of Lisbon
[2] Artificial Life Laboratory of the Department of Zoology, Karl-Franzens University, Graz, Austria
[3] Laboratory for Robotics and Intelligent Control Systems, Faculty of Electrical Engineering and Computing Zagreb, Croatia
rob.mills@fc.ul.pt

Abstract

We aim to better understand collective behaviours in social animals, doing so in the context of bio-hybrid societies, i.e., those that comprise robots and animals. Together, they make up a collective adaptive system, in which the self-organising patterns of the natural society can be understood, augmented and modified by the presence of robots. Here, we conduct a series of simulation-based experiments to investigate how natural behaviours in juvenile honeybees can be influenced by robots that are able to change key environmental stimuli. Firstly, we show specific couplings between animals and robots that can lead to symmetry-breaking and collective decision-making, even from an initially homogeneous environment. Secondly, we demonstrate that collective decisions made by animals in distinct habitats can be coordinated, through robots that share only relatively simple information-between habitats. Such mixed animal-robot societies exhibit multiple interactions and feedback loops, the understanding of which is key to the design of engineered parts that successfully harness the potential of the overall complex system.

Introduction

Social behaviours are abundant and intriguing, and by their very nature, dependent on a complex set of interactions with conspecifics as well as other environmental factors. Many social animals are crucial to human life (*e.g.*, pollination services from social insects; food from sheep) or can have devastating impact (*e.g.*, locust swarms). Thus better understanding what influences social behaviours is important for maintaining or enhancing 'beneficial' ones, and attenuate 'detrimental' ones. However, the complexity arising from interactions between individual behaviours can make the overall organisational outcomes difficult to understand. Individual-based modelling (Grimm and Railsback, 2005) and embodied in robots provide a method to test hypotheses about individual behaviours (Webb, 2001), as well as how those behaviours give rise to organisation at the collective level (*e.g.*, Campo et al., 2011). Integrating autonomous robots into animal societies is another approach to better understanding their behaviours (see *e.g.*, Halloy et al., 2007). Designing robots that interact with animals allows us to investigate what can influence the self-organisation processes involved in collective behaviours.

Our research program takes this approach. To verify that we have a proper comprehension of the animal societies and how to influence them, one of our overall goals is to coordinate the activities of multiple animal societies that occupy distinct habitats, facilitating communication between societies with robots that are accepted into each of the animal societies. This goal is ambitious, and here we use a simplified system to examine some more specific facets. In particular, we presently examine the interaction between multiple populations of the same species, and in this paper we present evidence from simulation modelling. We address two specific research questions: (i) can we identify conditions for mixed robot–animal societies in which joint decisions arise? (ii) can we coordinate such joint behaviours among animals occupying distinct habitats?

We use agent-based modelling to examine these research questions, focusing on the interactions in a system of honeybees and robots. We show feedbacks sufficient to provide symmetry breaking and joint animal–robot collective decision making: where the observed outcomes would not be reached by either the robot or animal populations alone. These results identify important interactions and feedback loops for self-organisation in such bio-hybrid societies, and additionally yield specific robot controller designs and interaction patterns that we will be able to empirically validate.

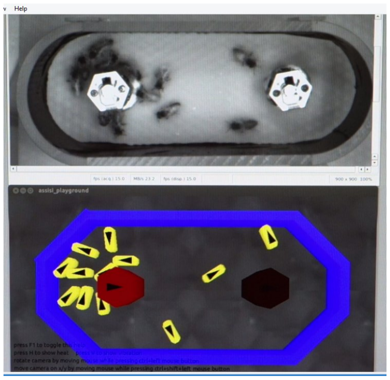
Self-organisation in social behaviours

For animals living in a group context, the social environment is key to survival. Moreover, many behaviours are exhibited at the level of the collective, such as group foraging and collective decision-making. Collective decisions are observed in many social species from primates (King and Cowlishaw, 2009) to social insects (Seeley et al., 2012), and can be driven by leadership or by self-organisation. In heterogeneous groups, *i.e.*, where individuals have variability in experience or abilities, leadership is frequently observed (King and Cowlishaw, 2009). Self-organisation can make use of active information transfer between individuals (such as the waggle dance in bees or pheromone trails in ants), but the modulation of individual behaviour without specific or active information transfer can also result in collective decisions (Jeanson et al., 2004). Self-organisation through reinforcement is seen in ant trail formation (Couzin and Franks, 2003), nest selection in honeybees (Seeley et al., 2012), and shelter selection in cockroaches (Amé et al., 2006).

Honeybees are an archetypal eusocial organism, whose group living exhibits various collective behaviours across their life stages, such as nest choice (Seeley et al., 2012). Recent work has shown that juvenile honeybees exhibit a preference for specific temperatures, and are collectively able to select between optima of differing quality in one environment (Szopek et al., 2013). This and other work shows that the decisions are socially mediated, *i.e.*, they only occur with a sufficient number of bees, and the more bees the faster the decision-making process is (see also Hahshold et al. (2011)).

Robotic swarms have been used to test hypotheses about specific mechanisms that lead to collective behaviours. Kernbach et al. (2009) study an aggregation mechanism based on honeybee behaviour; Garnier et al. (2005) investigate cockroach aggregation (and also argue that it exhibits a collective ability to 'measure aggregation site size'); and May et al. (2006) study aggregation in robotic rat pups and real animals. Gauci et al. (2014) aim to understand minimal requirements of the individuals that can result in aggregation. These works provide plausible distributed mechanisms for decision-making through aggregation, using only simple sensory and cognitive apparatus. Garnier et al. have studied cockroach shelter selection in homogeneous (2005; 2008) and heterogeneous (2009) environments. Campo et al. (2011) also consider how a similar robot swarm can select among sites of varying quality, but in addition, they show that the swarm will switch decision if a new site of higher quality is introduced. Here, the use of dynamic environments is of particular interest for our present work, although note that the dynamism is exogenous and not one that arises as a consequence of the animals' actions.

Rather than an experimenter hand-design the environmental dynamics directly, using robots to provide environmental changes can be responsive to how the animals act; this is an appealing approach to ethology, and collective behaviours in particular (De Schutter et al., 2001), since frequently the individual behaviours are relatively simple and therefore more feasible to implement in robotics (unlike, for instance, a complex mating ritual). Nonetheless, the connection between individual and collective behaviours remains challenging, and being able to substitute some individuals for robots without affecting the collective-level behaviour is a good indicator of model suitability (De Schutter et al., 2001). Vaughan et al. (2000) developed a robot sheepdog that was able to herd ducks; Faria et al. (2010) present a programmable fish robot and used it to investigate leadership behaviour. Halloy et al. (2007) developed robots that were accepted into a society of cockroaches, demonstrating that the group behaviours (shelter selection) are not modified when some animals are substituted by robots. Additionally, they exposed another facet of hybrid societies as an ethological tool: they reprogrammed the robots to favour the opposite choice from the insects, and showed that the robots were able to change the overall collective decisions, even when they were in the minority. This experiment highlights the interplay between social and environmental factors at work in self-organising collective processes, such as shelter selection.



press F1 to toggle this help
press H to show heat press V to show vibration
rotate camera by moving mouse while pressing ctrl+left mouse button
move camera on x/y by moving mouse while pressing ctrl+shift+left mouse button

Figure 1: Preliminary experiments with a hybrid society of robots, real bees, virtual bees and simulated robots.

Concept and preliminaries

Our research program revolves around bio-hybrid societies, integrating two prototypical species that reside in very different habitats – bees and fish as the primary example. We have developed custom robots that interact with each of the animals, and in the case of the bees, our robots are able to manipulate environmental variables in ways relevant to honeybees and to sense the local presence of bees (Griparic et al., 2015). Here we concentrate on the thermal environment (other stimuli are under investigation elsewhere). Juvenile honeybee groups are sensitive to the thermal environment: they remain 'paused', or in an aggregation, for longer when the local environment is warmer.

We have carried out some preliminary experiments that integrated real and simulated habitats, in which real bees and simulated bees both made collective choices and the robots shared information between the two habitats (Figure 1). These preliminary experiments form the basis of the present simulation study. In brief, these experiments used a binary choice assay in each arena, where each robot has to 'compete' to attract the bees, by varying its local temperature. The robots increase their temperature according to the number of bees nearby, providing a positive feedback loop. In this way, each animal-robot society together selects one of the robots as a winner. Moreover, we coupled pairs of robots (one in the real habitat and one in the simulated habitat), meaning that each robot would behave as if the sensors on its partner were its own, thereby reacting to both simulated and real bees. Consequently, the behaviours of the simulated bees influence the environment, and hence behaviours, of the real bees, and vice versa, despite the fact that the two groups reside in distinct habitats.

The following sections describe our simulation work that aims to better understand scenarios that merit empirical validation and further investigation.

Methods

This section details the models used to simulate the bees and robot controllers that make up our bio-hybrid society, as well as describing the available functionality in the hardware robots that we base the simulations on.

Bee behavioural model

We model the bees as autonomous, mobile agents whose behaviour is given by the "Beeclust" model (Schmickl et al., 2009). Inspired by observations of honeybees, the model features non-linear feedback loops that enable group-level aggregations to emerge. Honeybees form aggregations more rapidly in greater densities of conspecifics (and exhibit a threshold effect, *i.e.*, not aggregating below a certain density), and do so preferentially in regions closer to 36°C.

Beeclust comprises two main phases: random exploration, and pausing; as well as an obstacle avoidance state that can interrupt the exploration. When a bee encounters another bee it enters the pausing state, and the duration of a pause is proportional to the quality of the environment at the location of the contact. Specifically, since the present work considers thermal stimulus, a bee pauses for longer when it is warmer. Since the pause durations increase in warmer environments, and paused bees increase the local likelihood of others meeting them, positive feedback of these two factors can result in the formation and growth of aggregations.

We model bees that can discriminate between conspecifics and inanimate obstacles at close proximity, using infra-red (IR) sensors that provide a distance d and object type y. The bees have three sensors l, f, r that are oriented at -45°, 0, +45° relative to the bee's bearing. The behaviour is defined as follows.

```
loop:
1. delay(dt)
2. $((y_l, y_f, y_r), (d_l, d_f, d_r)) \leftarrow$ read_sensors($l, f, r$)
3. if $\exists i, (d_i < 0.5 \wedge y_i = \text{bee})$, for $i \in [l, f, r]$,
   (a) stop()
   (b) $T \leftarrow$ measure temperature
   (c) $t_{wait} \leftarrow$ compute_wait($T$)
   (d) sleep($t_{wait}$)
   (e) turn()
4. else if $\exists i, (d_i < 0.5 \wedge y_i = \text{obstacle})$, for $i \in [l, f, r]$,
   • turn()
5. else:
   • forwards()
end loop
```

The function `compute_wait(T)` uses data observed in juvenile honeybees (collected and analysed by Univ. Graz), and fitted with a hill function (similar to a sigmoid):

$$t_{wait}(T) = \left(\frac{(a + b \cdot T)^c}{(a + b \cdot T)^c + d^c} \right) \cdot e + f, \tag{1}$$

where $a = 3.09$, $b = -0.0403$, $c = -28.5$, $d = 1.79$, $e = 22.5$, and $f = 0.645$. The function yields low waiting times ($\sim$ 1 s) for T< 25 and high waiting times ($\sim$ 25 s) for T = 38°C.

Beeclust is a model based on honeybee behaviour, but it shares some features with other models of insect aggregation. The cockroach model in Garnier et al. (2005) involves a correlated random walk and a stopping state, which is dependent on the number of other *stopped* conspecifics nearby. The stopping duration is broken into short and long, which are each modulated by the local density. In Garnier et al. (2009), the transition to stopping is also modulated by the environment, using a binary threshold to the light level. Campo et al. (2011) extend this cockroach-based model to incorporate a quorum-sensing mechanism seen in ants to provide density estimation.

Aggregation is socially mediated in these models (including Beeclust). In principle, we could put forward alternative behavioural models that exclude social factors entirely, the aggregations instead resulting simply from a correlation among individual decisions (one option for Beeclust would be to use a random spontaneous transition to pausing in step 3). While such a model might be able to generate similar system-level outcomes for groups of bees, they depart from empirical observations of individual bees: when single (juvenile) bees are placed in an environment with differing temperatures in different zones, the majority do not choose any specific location (Szopek et al., 2013; Kernbach et al., 2009). Accordingly, we here use social models to develop useable predictions about social behaviour in real honeybees.

Robot interactions

We use custom-designed robotic devices that have can generate several stimuli with different modalities that the animals are sensitive to (*e.g.*, heat, light), and additionally the robots can sense several aspects of the environment (*e.g.*, temperature, IR) (Griparic et al., 2015). The robots have proprioceptive sensors that are used within the low-level actuator control: for instance, a Peltier device is used to generate heat. In a similar way to controlling the velocity of wheeled robots by setting the wheel speed, we set a temperature that the robot should produce. The robots are sessile, occupying fixed positions within the experimental arenas to interact with the animals.

In the present simulation study we use the robots' thermal actuators and 6 IR proximity sensors. In addition, the robots can communicate with specific neighbours (the topologies used are described with each experiment). We use the following program to define the temperatures over time.

At initialisation: set memory vector m of length m_{max} to zero for elements $m[0]..m[m_{max}-1]$.
For each timestep, t:

```
1. $d_{raw} \leftarrow$ count IR sensors above their threshold
2. $m[\mathrm{mod}(t, m_{max})] \leftarrow \texttt{saturate}(d_{raw})$
3. $\texttt{send}(\widehat{m}, \texttt{neighbours})$
4. if $\mathrm{mod}(t, t_{update}) = 0$:
    (a) $\mathbf{d}_x \leftarrow$ receive message(s) from $\texttt{neighbour(s)}$
    (b) $T_{new} \leftarrow \texttt{density\_to\_heat}(\widehat{m}, \mathbf{d}_x)$
```

where d_{raw} is a raw estimate of local bee density, $\widehat{m}$ is the mean value of vector m (*i.e.*, a time-averaged density estimate), $\mathbf{d}_x$ is a vector of density estimates received from neighbouring robots. In the topologies tested in this paper, robots have zero, one, or two neighbours.

Here we define the saturation function for the IR detection count as: $\texttt{saturate}(s) = \min(4, s)$. The density to heat function provides a linear transformation, mapping the time-averaged detection count to an output temperature. The function is parameterised to allow for different topologies examined. For each robot x involved there is a contribution, $c_x = \frac{\widehat{d_x}}{4}(T_{max} - T_{min})$ that depends on a robot's temperature. These are then combined as a weighted sum:

$$T_{new} = T_{min} + \sum_{x \in \{l,r,c\}} c_x w_x, \tag{2}$$

where the relative weights of each robot's contribution depends on the specific setup (defined with each experiment).

Simulation environment

We simulate the interaction of bees and robots using a real-time platform for 2D robot simulation[1], with customisations specific to the form of experiments, environments, and agents involved in our research[2]. Specifically, heat, vibration, and light stimuli are modelled, and the robot models include actuators and/or sensors for these modalities as appropriate (*e.g.*, the juvenile honeybee is sensitive to heat, but does not produce heat itself). The software is designed such that the exact same code that runs the simulated robots can be transferred directly to the hardware counterparts.

Quantification of collective decisions

The experiments below take the form of a binary choice assay, allowing us to assess whether the bees have reached a collective decision using statistical tests. Specifically, all of the bee positions at the end of the experiment are divided into two zones of equal size (*i.e.*, closer to one robot/heat source or the other). Then, following (Halloy et al., 2007; Szopek et al., 2013), we use a binomial test with prior probabilities of 0.5, and a significant deviation from this uniformly random distribution is considered a collective decision.

Simulation experiments

This section describes the three experiments that we conduct to investigate our bio-hybrid society. 1) We start with an environment in which the robots act simply as heat sources, establishing a gradient in the temperature that the bees experience. This serves to verify whether our model behaves as per laboratory experiments that examined natural behaviours of honeybees. 2) We close the loop between robots and animals by using robots that are sensitive to the bees, such that the actions of the animals influence the actions of the robots, and vice versa. We examine how different forms of feedback influence the self-organisation of this overall system. 3) We examine a dual-arena scenario, where there is coupling across the arenas via the robots. Here, the bees are separated into two sub-populations that are spatially distal, and thus cannot interact directly. However, by interacting with the robots in their own environment, each sub-population of bees can influence the other sub-population.

Parameter settings common to all experiments

Many of our choices are motivated to mirror conditions that we have used in our animal-based experiments, including the arena and robot setup, the temperature ranges that are relevant and not harmful to the animals. This will facilitate more straightforward transfer to empirical validation.

Throughout the paper, $T_{min} = 28$°C and $T_{max} = 38$°C; the ambient temperature is 27°C. The modelled bees measure 13.5 mm × 5 mm (based on our measurements), and detection range is 5 mm. Memory length $m_{max} = 18$, $t_{update} = 3$ s, $t_{resample} = 0.5$ s. (*i.e.*, the current update is based on the previous 6 cycles). As noted in the methods, we take a simple time average across the memory. Preliminary experiments with other windows (*e.g.*, Hamming) did not provide more accurate estimates and thus we use the simplest square window here. The arenas used are rectangular with rounded ends (see Figure 1). They have dimensions 210×65 mm (internal). The two robots are positioned 60 mm either side of the centre, on the midline. Bees are initialised with random position and orientation.

[1] Enki – an open source fast 2D robot simulator http://home.gna.org/enki/

[2] ASSISI playground – an open-source simulator of mixed societies https://github.com/larics/assisi-playground

Temperatures	5 bees	10 bees	12 bees	15 bees
28°C/30°C	0	2	3	1
28°C/32°C	0	3	9	14
28°C/34°C	0	5	17	19
28°C/36°C	0	15	17	25
28°C/38°C	0	20	22	27

Table 1: Frequency of repeats (out of 30) in which a collective decision was reached at the end of the experiment, with two fixed heat sources. Collective decisions are reached more reliably with warmer temperatures and with more bees.

Symmetry-breaking in honeybees can be mediated by a heterogeneous environment

In this experiment we set up a plain binary choice for the virtual honeybees, by providing two zones that are warmer than the ambient environment. The temperatures of these two zones are kept constant throughout an experiment, achieved by fixing the set-point of each robot's temperature output (*i.e.*, the robots act simply as fixed heat sources in this case). We use this scenario to assess whether our simulated honeybees express similar collective preferences to those observed in natural honeybees, and do so by varying the number of bees and the temperature that the warmer heat source is set to (we elect the East source to be hotter).

Experiments are 15 mins each, and we perform 30 repeats for 5 temperature pairs, for each of 4 group sizes (5, 10, 12, 15 bees). In each repeat, we measure whether the bees have reached a collective decision using a binomial test as described above. The results for a significance level of $\rho < 0.05$ are shown in Table 1. This data shows that collective decisions are reached significantly more in higher temperatures and with larger group sizes. If the group size is too small, collective decisions are never reached. We are therefore in a position to examine more complex scenarios that employ active robots.

Symmetry-breaking and joint collective decisions

In this set of experiments, we examine how bees interact with adaptive robots, and under what conditions joint collective decisions are reached by the bio-hybrid society. We introduce a coupling between the local bee density around a robot and the temperature that the robot sets its local environment to. The robots start out with the same (low) level of attraction to the bees, but the bees' actions can manipulate it such that specific regions become more preferable.

The two robots in any experiment use the same controller. The robot refers to itself as l (local) and the other robot as c (competitor). We examine two robot controllers: BI uses local excitation only ($w_l = 1$, $w_c = 0$); while BII uses local excitation and cross-inhibition ($w_l = 1$, $w_c =$ -0.5).

We provide results for these two controllers, and two further control conditions that use fixed heat sources: (i) with a single strong attractor (28/38°C, SA); and (ii) with two strong attractors (38/38°C, TA). Above, we saw that higher temperatures induce stronger collective decisions (hence we compare with control SA); but we are also interested in unbiased decision-making rather than exogenously asserting a preferred outcome, hence the inclusion of control TA. Our goal is to verify that the active sensori-actuator loop has a significant impact.

Controller	W	E	Decision (W or E)	No decision
BI	32	27	59	41
BII	35	41	76	24
SA (fixed 28/38°C)	0	88	88	12
TA (fixed 38/38°C)	16	13	29	71

Table 2: Number of independent runs that reached a collective decision with $\rho < 0.05$, for two active controller types, and fixed conditions for comparison.

We use 15 bees in all experiments of 15 mins, and obtain 100 repeats for each scenario. We apply the binomial test for collective decision-making as above, reported in Table 2 (also showing which side the bees decided upon). We also provide a more detailed view of the outcomes in Figure 2. Here, we compute the proportion of bees that are in each of the two zones for each 10 s during the final quarter of the experiment, and record the mean value for each run.

To analyse whether the decisions reached are biased, we compare the runs where the bees chose to the left versus those that chose to the right. With an unbiased null hypothesis, we find that neither active controller, nor TA, is biased (binomial test, $\rho > 0.05$), but the fixed control SA is significantly biased ($\rho < 0.05$). Considering the rate of collective decision-making, both active controllers achieve a rate significantly higher than TA (χ^2 test, $\rho < 0.05$), and using cross-inhibition (BII) significantly increases the rate compared with BI (χ^2 test, $\rho < 0.05$).

Firstly, our results show that the self-organised systems exhibit collective decisions, from an initially homogeneous environment. That is, the bees have induced conditions that they favour, and aggregated in that region as a result. This is qualitatively different from when fixed heat sources exogenously exerted a particular outcome in the bee population. Accordingly, we refer to this outcome as a joint animal-robot decision. Secondly, while it is relatively straightforward to induce a decision that is known *a priori* (by simply turning one heat source to maximum), for a hybrid society to self-organise reliably is not so simple. If the two robots were to act 'selfishly', we might expect them to both maximise their attractive stimulus, and we see that this causes both of them to lose in the majority of cases. Conversely, using a combination of local positive feedback and cross-inhibition achieves a high level of collective decision-making, which, moreover, are not biased *a priori* to any specific decision.

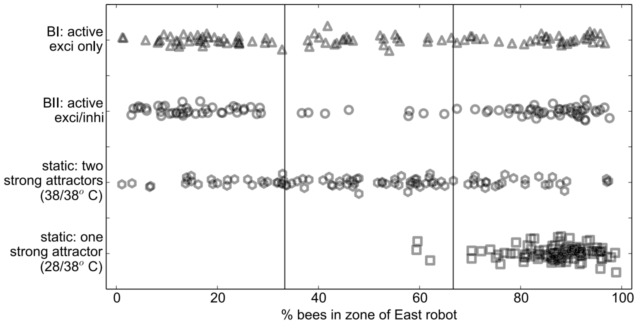


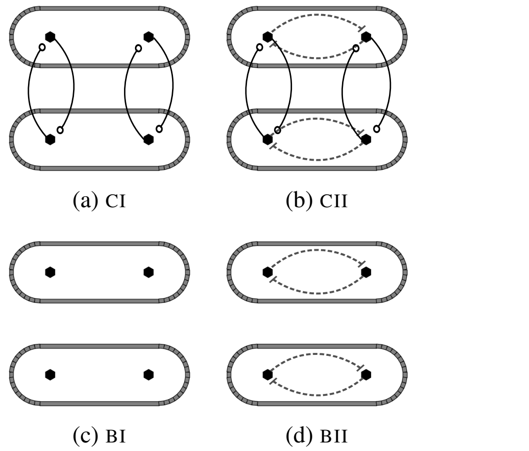
Figure 2: Location of bees during final quarter of each experiment, showing the mean percentage of bees in the East zone, with one point per experiment. (A small amount of jitter is used on the y-axis, as well as a low-alpha value, to better visualise the density of the distributions). With a single strong attractor, the bees select the hotter robot in all repeats; whereas with two strong attractors the bees fail to choose in the majority of cases. Comparing the two active controllers, we observe that there is no bias towards one robot or the other: the overall system makes joint collective decisions that are not achieved without the interactive robots.

Coordinating decision-making in distal populations

We extend the previous scenario to include multiple local habitats, each containing two robots and a population of bees. In each arena, bees and robots can interact with one another, but between arenas no proximal/physical contact is possible. Thus, the bees are unable to directly interact with bees in other populations. However, we establish specific inter-robot links, over which robots share their local bee estimates between the arenas (see Figure 3). In this way, bees in one arena can *indirectly* influence local environments in distal arenas. We use these scenarios to investigate how the bees and robots interact, within and between arenas. We aim to assess whether the collective decisions reached in one arena can be coordinated with those reached in a distal arena.

We examine variants of the two controllers as used in the previous experiment, extending to include a remote (r) interaction as well as the local and competitor interactions: CI uses excitation from local and remote robots ($w_l = 0.5$, $w_r = 0.5$, $w_c = 0$); CII also uses cross-inhibition ($w_l = 0.5$, $w_r = 0.5$, $w_c =$ -0.5). As control experiments, we disable inter-arena communication ($w_r = 0$), *i.e.*, the same controllers BI and BII from above. The control experiments provide a baseline and allow us to assess whether the explicit inter-arena robot–robot communication is responsible for any coordination that arises. We carry out 50 independent repeats for each controller/topology, each using 15 bees per arena, and lasting for 15 mins.

Figure 4 shows the distribution of the bee locations at the end of each run. We treat each arena separately, dividing the area into two zones, West and East. We calculate the fraction of bees in each zone every 10 s, and take the mean over the last 120 s, yielding four values for each experiment. However, each arena state is described by one fraction, allowing us to plot the outcome of each run with two dimensions.


(a) CI (b) CII

(c) BI (d) BII

Figure 3: Topologies tested in the multi-arena experiments. Solid lines indicate positive contributions and dashed lines indicate negative contributions to the receiving robot.

Controller	WW	WE	EW	EE	CD	CCD	χ^2
CII	14 (19)	0	0	15 (20)	29	39	25.1 (*)
CI	9 (17)	0	0	6 (10)	15	27	11.1 (*)
BII	7 (10)	8	10	3 (7)	28	17	1.56 (n.s.)
BI	3 (7)	4	4	2 (11)	13	18	0.09 (n.s.)

Table 3: Distribution of collective decisions reached. Values in brackets treat all 30 bees as one group. CD = frequency of collective decisions reached in both arenas separately; CCD = coordinated collective decisions reached, *i.e.*, in both arenas jointly.

In addition we quantify the collective decisions made, firstly, by each group of bees separately, and then lumping both together as if they were a single group. We apply a binomial test to the number of bees in each zone at the end of the experiment, to assess whether the group has made a significant decision. Table 4 reports these values for $\rho < 0.05$.

To quantify whether the decisions are coordinated between arenas, we perform the following statistical tests that are based on the collective decisions in each arena. Firstly, we test whether the decisions made in each arena depend on one another, using a χ^2 test with the null hypothesis that the outcomes in each arena are independent. This confirms that there is a strong correlation in the outcomes of the two arenas with controllers CII and CI, but that, as expected ($\rho < 0.05$), the outcomes do not coordinate when the inter-arena links are absent ($\rho > 0.05$). Secondly, where inter-arena coordination emerges, there is no bias towards either of the outcomes (binomial test, $\rho > 0.05$ for both CI and CII). Where coordination does not emerge, there is no significant

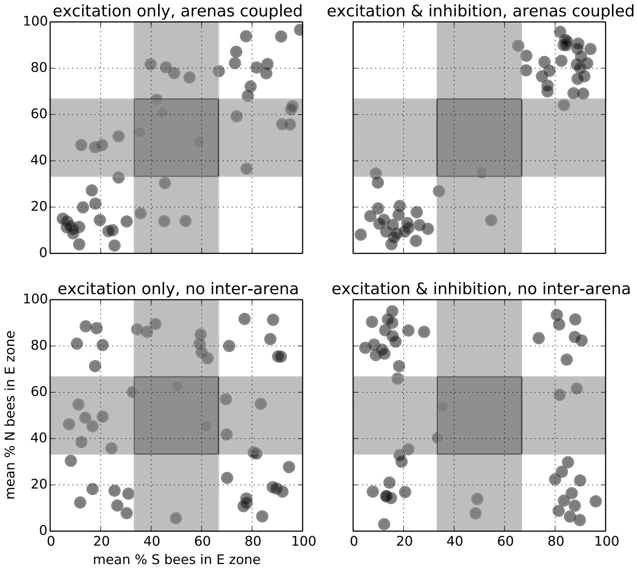


Figure 4: The location of bees during the last 120 s of each experiment, for the four controllers. We observe that: [top row] (1) including inter-arena communication is sufficient to coordinate decisions; (2) there is no *a priori* bias to one coordinated outcome or the other; [top versus bottom] (3) without inter-arena communication, all four outcomes occur; [left versus right] (4) cross-inhibition substantially improves the strength of decisions.

bias to any of the four outcomes (χ^2 test with equal priors, $\rho > 0.05$, for both BI and BII). Thirdly, we compare the overall level of decision-making in each condition. Cross-inhibition significantly affects the rate of decision making in CII over CI (χ^2 test, $\rho < 0.05$ for both CD and CCD). The inter-arena links do not have a significant influence on the rate of collective decisions when considered in each arena independently (χ^2 test for CD, $\rho > 0.05$ for both BI to CI and BII to CII). However, when considering the decisions together (CCD), the inter-arena links have a significant effect (χ^2 test, $\rho < 0.05$ for both BI to CI and BII to CII).

To summarise, these results show the following: (i) strong correlation in the outcomes of the two arenas with controllers CII and CI, but that, as expected, the outcomes do not coordinate when the between-arena links are absent. (ii) the inter-arena communication significantly increases the rate of coordinated collective decision-making, but does not significantly alter the rate of decision-making within the individual arenas. (iii) from the two combinations of decisions observed in the two arenas, there is no bias towards one outcome or the other. It is important to note, however, that with controllers CI and CII, since the populations do not act independently, the overall outcomes are constrained to a subset of the possible collective decisions that are made in the absence of the inter-arena coupling.

These results show, therefore, that the interwoven feedbacks and interactions between bees and bees, bees and robots, and robots and robots, can result in coordination of multi-species collective decisions, even among spatially distal arenas.

Discussion and conclusions

In this paper, we used agent-based modelling to investigate how bio-hybrid societies make joint animal–robot collective decisions, The first experiment confirmed that when a population of bees (simulated with the Beeclust model) experiences a heterogeneous environment, they make collective decisions that favour warmer regions, and that the frequency of decision-making depends on the population size and temperature of the warmer region. The second experiment addressed our first research question by showing that positive feedback between bee presence and the heat that the robots emit can result in collective decisions, and that including cross-inhibition significantly strengthens their frequency. Moreover, since the robots are initially equally attractive to the bees, we have no *a priori* expectation for which robot the bees will choose. Rather, the final outcome arises due to the joint action of both robots and animals.

The third experiment demonstrated that a coupling of the sensory information of robot pairs that occupy distinct environments is sufficient to coordinate the decisions made by populations in those separated environments, even though the animals cannot directly interact. This provides a positive solution to our second research question regarding coordination of collective decisions among separate habitats. We have also shown that that fewer links are sufficient to coordinate decisions across arenas (Mills and Correia, 2015).

The coordination shown in the third experiment is facilitated with relatively simple and abstract information. While here we used homogeneous robots to mediate the coordination, different agents could interpret the information differently in order to interact with different species. Such a system may allow for combinations of engineered and natural devices that would not otherwise be compatible (due to mismatches in spatial or temporal scales, or lack of appropriate sensors and actuators).

Self-organisation is one of the main classes of mechanism that achieves adaptation in biological systems, and can operate at the collective level (cf. individual learning) during the lifetime of the individuals (cf. evolution). Our mixed bio-robot ecosystem also comprises adaptive mechanisms on population and individual levels, both positive and negative feedback loops. These interwoven feedbacks and interactions result in organisation at the system level, operating at local or spatially distributed scales within the whole population of robots and bees. The adaptation-generating feedback loops emerge from interactions among multiple components of the system: one positive and two negative feedback loops. The positive feedback loop involves the coupling between bees waiting time, the temperature of a locale, the detection of the bees by a robot, and the positive correlation of the heat emitted by the robot. Complementarily, the warmer it is near a robot, the more energy flows away due to diffusion, creating a negative feedback loop. These feedback loops allow robots to adapt such that they achieve a certain maximal local density of bees around them. However, since the bees are a shared and limited resource within each arena, the robots compete for the bees. This competition is augmented by the cross-inhibitory signals between robots. Collective consensus is achieved by the parallel localised positive feedback loops while stability of the collective system arises from the negative feedback loops.

In this simulation study we have developed specific robot controllers. The co-design of our simulator and robot hardware will facilitate transfer of these controller programs to the hardware, allowing us to empirically validate our findings in mixed robot–animal–simulated agent societies. In future work we also aim to explore more diverse tasks that the overall societies will face, including where environments undergo exogeneous perturbations in addition to the dynamics introduced by the robots, and tasks where populations with different characteristics will bring their strengths to collaboratively find solutions.

Acknowledgments

This work is supported by: EU-ICT project "ASSISI|bf" no. 601074, and by centre grant (to BioISI, ref: UID/MULTI/04046/2013), from FCT/MCTES/PIDDAC, Portugal.

References

Amé, J.-M., Halloy, J., Rivault, C., Detrain, C., and Deneubourg, J. L. (2006). Collegial decision making based on social amplification leads to optimal group formation. *PNAS*, 103(15):5835–5840.

Campo, A., Garnier, S., Dédriche, O., Zekkri, M., and Dorigo, M. (2011). Self-organized discrimination of resources. *PLoS ONE*, 6(5):e19888.

Couzin, I. D. and Franks, N. R. (2003). Self-organized lane formation and optimized traffic flow in army ants. *P Roy Soc Lond B*, 270(1511):139–146.

De Schutter, G., Theraulaz, G., and Deneubourg, J.-L. (2001). Animal–robots collective intelligence. *Ann Math Artif Intel*, 31(1-4):223–238.

Faria, J. J., Dyer, J. R., Clément, R. O., Couzin, I. D., Holt, N., Ward, A. J., Waters, D., and Krause, J. (2010). A novel method for investigating the collective behaviour of fish: introducing 'robofish'. *Behav Ecol Sociobiol*, 64(8):1211–8.

Garnier, S., Gautrais, J., Asadpour, M., Jost, C., and Theraulaz, G. (2009). Self-organized aggregation triggers collective decision making in a group of cockroach-like robots. *Adapt Behav*, 17(2):109–133.

Garnier, S., Jost, C., Gautrais, J., Asadpour, M., Caprari, G., Jeanson, R., Grimal, A., and Theraulaz, G. (2008). The embodiment of cockroach aggregation behavior in a group of micro-robots. *Artif Life*, 14(4):387–408.

Garnier, S., Jost, C., Jeanson, R., Gautrais, J., Asadpour, M., Caprari, G., and Theraulaz, G. (2005). Collective decision-making by a group of cockroach-like robots. In *Swarm Intelligence Symposium*, pages 233–240.

Gauci, M., Chen, J., Li, W., Dodd, T. J., and Gro, R. (2014). Self-organized aggregation without computation. *Int J Robot Res*, 33(8):1145–1161.

Grimm, V. and Railsback, S. F. (2005). *Individual-based modeling and ecology*. Princeton university press.

Griparic, K., Haus, T., Bogdan, S., and Miklic, D. (2015). Combined actuator sensor unit for interaction with honeybees. In *Sensor applications symposium (to appear)*.

Hahshold, S., Szopek, M., Thenius, R., Schmickl, T., and Crailsheim, K. (2011). How modulation of resting time affects collective decision making in honeybees. *Apimondia*.

Halloy, J., Sempo, G., Caprari, G., Rivault, C., Asadpour, M., Tâche, F., Said, I., Durier, V., Canonge, S., Amé, J. M., et al. (2007). Social integration of robots into groups of cockroaches to control self-organized choices. *Science*, 318(5853):1155–1158.

Jeanson, R., Deneubourg, J.-L., Grimal, A., and Theraulaz, G. (2004). Modulation of individual behavior and collective decision-making during aggregation site selection by the ant messor barbarus. *Behav Ecol Sociobiol*, 55(4):388–394.

Kernbach, S., Thenius, R., Kernbach, O., and Schmickl, T. (2009). Re-embodiment of honeybee aggregation behavior in an artificial micro-robotic system. *Adapt Behav*, 17(3):237–259.

King, A. J. and Cowlishaw, G. (2009). Leaders, followers, and group decision-making. *Commun Integr Biol*, 2(2):147–150.

May, C. J., Schank, J. C., Joshi, S., Tran, J., Taylor, R., Scott, I., et al. (2006). Rat pups and random robots generate similar self-organized and intentional behavior. *Complexity*, 12(1):53–66.

Mills, R. and Correia, L. (2015). The influence of topology in coordinating collective decision-making in bio-hybrid societies. In *Procs. EPIA (to appear)*.

Schmickl, T., Thenius, R., Moeslinger, C., Radspieler, G., Kernbach, S., Szymanski, M., and Crailsheim, K. (2009). Get in touch: cooperative decision making based on robot-to-robot collisions. *Auton Agent Multi Agent Syst*, 18(1):133–155.

Seeley, T. D., Visscher, P. K., Schlegel, T., Hogan, P. M., Franks, N. R., and Marshall, J. A. R. (2012). Stop signals provide cross inhibition in collective decision-making by honeybee swarms. *Science*, 335(6064):108–111.

Szopek, M., Schmickl, T., Thenius, R., Radspieler, G., and Crailsheim, K. (2013). Dynamics of collective decision making of honeybees in complex temperature fields. *PLoS ONE*, 8(10):e76250.

Vaughan, R., Sumpter, N., Henderson, J., Frost, A., and Cameron, S. (2000). Experiments in automatic flock control. *Robot Auton Syst*, 31(1):109–117.

Webb, B. (2001). Can robots make good models of biological behaviour? *Behav Brain Sci*, 24(06):1033–1050.

Corpus-taught Evolutionary Music Composition

Csaba Sulyok[1], Andrew McPherson[1], Christopher Harte[2]

[1]Queen Mary University of London
[2]University of York
csaba.sulyok@gmail.com

Abstract

In this paper we present a music composition system that uses a corpus-based multi-objective evolutionary algorithm. We model the composition process using a Turing-complete virtual register machine to render musical models. These are evaluated using a series of fitness tests, which judge the statistical similarity of the model against a corpus of real music. We demonstrate that the methodology succeeds in creating pieces of music that converge towards the properties of the chosen corpus. These pieces exhibit certain musical qualities (repetition and variation) not specifically targeted by our fitness tests; they emerge solely based on the similarities.

Introduction

The process of creating and assessing music is fundamentally subjective and hard to define. Composers spend years perfecting and evolving their technique; to create new music successfully, both experience of the composition process and knowledge of existing 'good' music is required. The composers judgement as to whether a new musical idea is good or bad will be a subjective decision based on their knowledge and memory of previous pieces (see Figure 1). As their creative process evolves, so too should the quality of their compositions.

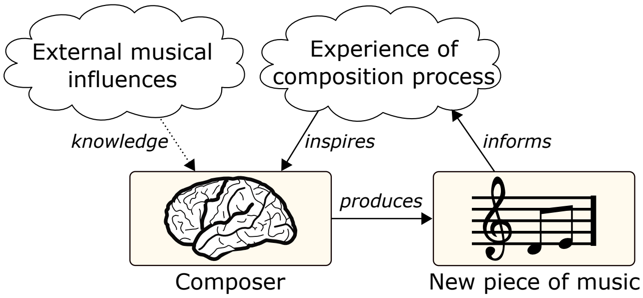

External musical influences
Experience of composition process
knowledge
inspires
informs
produces
Composer
New piece of music

Figure 1: The creative process of a composer is informed by both experience of music in general (external influence of others' music) and experience gained through practice of the composition process itself.

Since the development of a composer's skill can be viewed as an evolving process, it seems intuitive that evolutionary computation techniques should find utility in algorithmic music composition. Using evolutionary models for composing music is nothing new; examples date back to the early 1990s (Hartmann, 1990; Gibson and Byrne, 1991; Horner and Goldberg, 1991; Biles, 1994; McIntyre, 1994). However, the subjective nature of music quality makes defining appropriate machine fitness tests difficult (Miranda and Biles, 2007; Waschka, 2007). As a result, much previous research in this area has relied on some degree of human feedback for fitness evaluation (Hartmann (1990); Jacob (1995); Tokui and Iba (2000); for an exhaustive review see Rodriguez and Vico (2014)).

To reduce the otherwise vast solution space of all possible music, many previous approaches have focused on evolving one particular musical property such as rhythm patterns (Tokui and Iba, 2000; Eigenfeldt, 2009; Martins and Miranda, 2007)), melody (Towsey et al., 2001) or harmonisation (Phon-amnuaisuk et al., 1999). Other work looked at generating small musical motifs (*GenDash*, Waschka (2007); *SBEAT*, Unemi (2002)) or evolving improvisatory passages (*GenJam*, Biles (2007)).

In more recent years, evolutionary composition has seen a rise in interest again: De Prisco et al. (2011) used multi-objective differential evolution to solve unfigured bass harmonization; Dostál (2012) explored autonomous fitness tests to evolve rhythm accompaniments; and MacCallum et al. (2012) created the online *DarwinTunes* community to crowd-source human feedback for choosing which music pieces will be selected for breeding.

When designing an evolutionary system, we must first answer the question: "What is this system supposed to evolve?". Previous composition systems have generally attempted to evolve musical pieces but in this paper we propose evolving the composition *process* instead. We cast the action of composing a piece of music as a process running on a Turing-complete virtual computing machine. The virtual machine has a set of instructions that will be executed in a given order depending on the initial state of its memory (i.e. its program) and some way of writing notes onto a musical ’score’ whenever an output instruction is encountered. The development of the composer’s skill then becomes a genetic programming problem; the genotype is the program string presented to the virtual machine and the phenotype its musical output. In such a system, the executing process on the virtual machine can hold internal structuring rules and information that are not visible in the final musical phenotype. Whorley et al. (2013) address this point in the context of melody harmonization, the exact steps of which cannot be known just by listening to the piece of music. Decoupling the genetic program and the resulting musical phenotype in this way allows for the emergence of complex structures not possible in more constrained musical models. For example, a conditional branch statement in the program can cause the repetition of a single note or section, or perhaps even the entire piece depending on where it occurs. While the musical possibilities of this approach increase in number, the resulting space of possible outcomes is very large and hard to analyse (McIntyre, 1994; Martins and Miranda, 2007) so careful design of fitness evaluation is necessary.

As mentioned above, the composer’s creative process cannot operate in a vacuum; it is necessarily dependent on the influence of works from previous composers. In the same way, our evaluation process employs fitness tests that judge the similarity between statistical properties of the current musical phenotype and those of a chosen corpus of existing music (in this case, a group of Bach keyboard exercises). Our evolutionary algorithm is multi-objective in nature, incorporating tests for a number of different properties (Fonseca and Fleming, 1993). Using statistical similarity with a corpus as musical fitness evaluation has been discussed before (Eigenfeldt (2012), Manaris et al. (2007)), but without the inclusion of a Turing-complete virtual machine.

Our aim here is to establish whether the use of virtual machine and corpus-based similarity tests together will help the system converge towards the properties of the chosen corpus. Although the phenotype in our current system is music, we propose this method as a generic one that can be applied to the evolution of any creative process.

System overview

Our system follows a conventional genetic programming structure (see Figure 2). An initial population of genotypes (genetic strings) is generated randomly then each individual is run on the virtual machine in turn. The output of the virtual machine is a byte stream that is parsed by a *model builder* to create our phenotype, a musical model. The structure of this model is completely independent of the structure and mechanism of the virtual machine. This two-stage approach to rendering the model deviates from previous musical evolutionary systems in that the genetic string is not directly used to build the phenotype, instead being *interpreted* by the virtual machine.

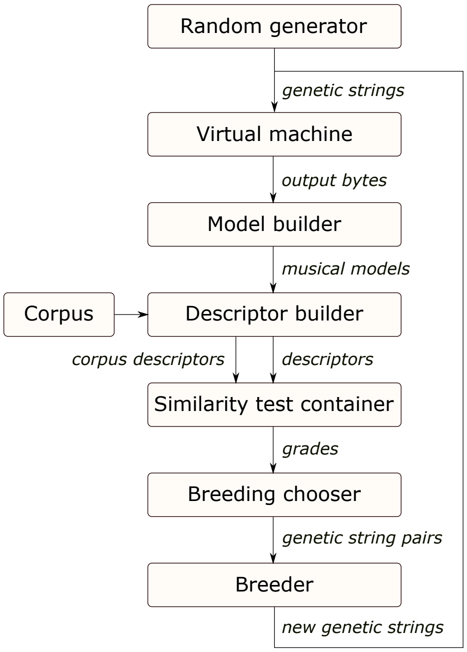


Figure 2: Workflow of the algorithm: A population of genetic strings is interpreted by the virtual machine and the resulting bytes used to build models; statistical transforms are performed on the models to yield descriptors, which are used in similarity tests to assess fitness of the musical pieces. Based on these grades, genetic strings are bred and mutated to produce the next generation.

Each phenotype model is evaluated using a series of tests derived from the corpus. These tests focus on the underlying statistical properties of a model rather than the similarity of the data itself. High grades can therefore be achieved not only when a model is identical to a member of the corpus, but also when it is statistically similar to them. To produce these statistics, we subject models to a series of transform methods to produce a *descriptor*. The construct which creates a descriptor from a musical model is a *descriptor builder*. We refer to the complete set of tests as the *similarity test container*.

Based on the assigned grades, the *breeding chooser* chooses pairs of units for crossover, and the *breeder* splices these pairs to create offspring. The crossover and mutation process operates on the genetic strings producing a new generation of programs for the virtual machine to execute.

Virtual machine

Our Turing-complete virtual machine interprets commands by reading 8-bit values from the current address pointed to by the the program counter and executing the assigned instructions. All possible values of a byte become potential opcodes mapped to a custom instruction set. The following

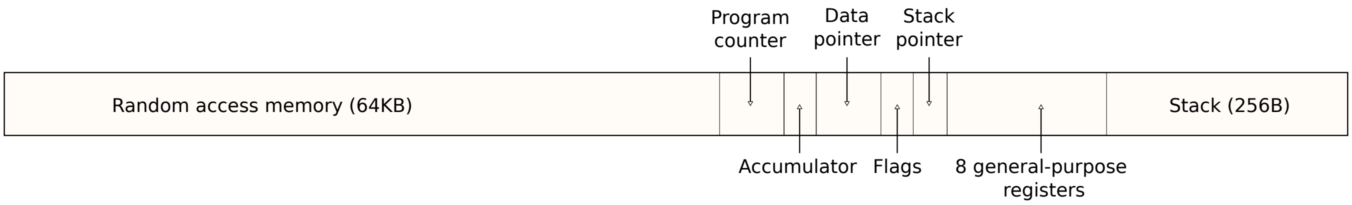


Figure 3: Regions of the genetic string dictate the complete initial conditions of the virtual machine. The first 64kB represents the machine's main memory; eight 8-bit general-purpose registers and an accumulator are provided for data manipulation; a 16-bit program counter and data pointer allow addressing of the main memory; a separate 8-bit stack pointer allows the machine to control a 256-byte call stack.

types of operations are defined:

1. *Data transfer* - copy a segment of the RAM or a register to another segment of the RAM or to a register;
2. *Arithmetic & Logic* - perform simple arithmetic/logical functions on a value within the RAM or a register;
3. *Branching & conditional instructions* - move the program counter to a different location, optionally based on a condition;
4. *Machine control* - internal commands such as halting or pushing/popping using the stack;
5. *Output commands* - outputs a value from the RAM or a register; this is the main customization we use to make the virtual machine serve our purpose.

The virtual machine interprets bytes until a 'halt' command is found or until it has processed a preset maximum number of commands or outputs.

The genetic string defines the initial condition of the virtual machine; it encompasses the initial value of the random access memory, the stack and all registers. When a genetic string is fed into the virtual machine, all these segments are set as shown in Figure 3.

Since our virtual machine is a Von-Neumann machine, it can overwrite its own memory while the process executes. To avoid the genetic string being changed in this process, it is always copied rather than moved in the feeding process. This ensures that interpreting the same genetic string multiple times will always result in the same output. Therefore we can safely save, recall or re-evaluate any genotype.

The musical model

Here, we define our musical model as a series of notes, each having three properties:

1. *Inter-onset interval (IOI)* - The time elapsed between the previous and current note onsets. For the first note, it is the time between the beginning of the piece and the note's onset. The unit of measurement for this property can be set globally; in this case, it is a 16th note value using a tempo of 120 BPM.

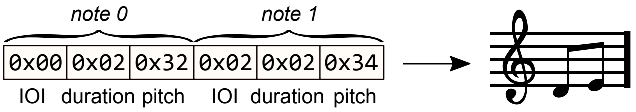


Figure 4: The model building process: parsing bytes to obtain note properties. The byte stream output from the virtual machine is sliced into three-byte chunks representing notes using one byte per property.

2. *Duration* - The current note's duration, in the same units as the IOI.
3. *Pitch* - A numeric value representing the note's pitch between 0 and 127, mapped to the same pitches as defined in the MIDI[1] protocol. The value 69 is associated with the 440Hz concert A, an increase or decrease of one unit representing a pitch rise or fall of one semitone respectively.

To build an instance of this model, the model builder parses the output of the virtual machine (see Figure 4). It masks each byte to give the boundaries of the three properties: Inter-onset interval and duration use 4 bits (longest note is the whole note), and pitch uses 7 bits (values are between 0 and 127). The model builder ignores any note whose duration or pitch is 0.

The resulting model is essentially a 3-row matrix, where each row represents a property, and each column represents a note. Given a model m, the matrix element $m_{2,n}$ is the pitch of the nth note. The model builder populates this matrix column-by-column.

Corpus-based fitness tests

The fitness of any rendered phenotype is determined by a series of similarity tests. All tests aim to evaluate how statistically similar a model is to the models in the corpus. This allows a large space of phenotypes to achieve high scores on the tests, not just the ones identical to a member of the corpus.

Here we take a holistic approach to building these tests, i.e. we test fundamental properties of our models instead

[1] Musical Instrument Digital Interface

of properties specific to music theory. We wish to find out whether looking only at the fundamental distributions' similarities allows the system to converge towards musical traits without the need to specifically test for them. This approach allows us to keep the generic aspect of this algorithm, so non-musical models may also benefit from it.

We define two types of tests: those revolving around a single property, such as total duration or overall number of notes, and those revolving around statistical property distributions.

We will refer to the overall number of tests as T. Given a model m and test index t, the test $f_t(m)$ returns a *grade* $g_{t,m}$, which is a numerical value between 0 and 1. The model's overall grade $\bar{g}_m$ is determined by a linear combination of the individual grades per test. We will refer to the coefficients of this equation as *importances*, since a higher coefficient means the test impacts the overall grade more. Every test f_t is assigned a predefined importance i_t. The importances are scaled to sum to 1 over all tests:

$$\sum_{t=0}^{T-1} i_t = 1, \qquad (1)$$

therefore the overall grades will also be between 0 and 1:

$$g_{t,m} \in [0,1] \quad \Rightarrow \quad \bar{g}_m = \sum_{t=0}^{T-1} i_t g_{m,t} \in [0,1]\,. \qquad (2)$$

Chosen corpus

We have chosen Bach's *Inventions and Sinfonias*, comprising 30 keyboard exercises, as our corpus. This catalogue of musical pieces was chosen for the following reasons:

- *Relatively short pieces of equal length* - For this research, we wish to test if, given a corpus of small length pieces, our system will converge to create pieces of similar length. Using this constrained length can help demonstrate that our algorithm finds the appropriate solution subspace where pieces of these lengths live.
- *Constant tempo* - We have not included tempo as a property which changes within the models (it is preset to 120 BPM) and we do not test for it.
- *Similar style* - Stylistically, all pieces of the corpus are similar in phrasing and property distribution. By using them, we once again limit our solution subspace and test whether or not the algorithm can find it.

The corpus is has been extracted from a set of MIDI files[2]. Our musical model has a structure similar to MIDI, therefore the algorithm has been extended to support conversion between the two representations.

The corpus files all comprise a single track of polyphonic music, therefore our system currently also produces single-track polyphonic music. The files in the corpus have been generated from a score rather then recorded by a human player. As a result they contain no *musical expression* such as changes in dynamics and tempo (e.g. all notes have velocity 127). For this reason our current systme does not use velocity as a property of notes within the models but it could easily be added at a later stage.

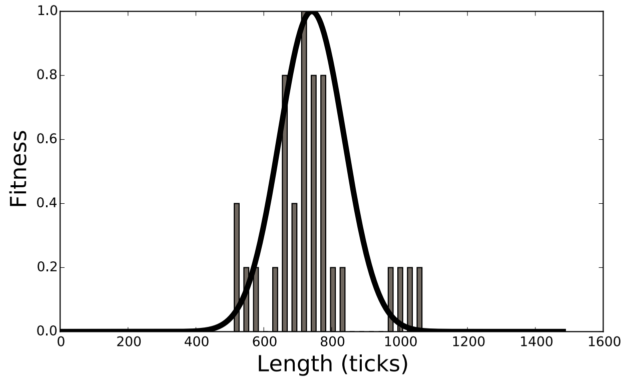


Figure 5: Deduced normal distribution test for musical model length. The length distribution for the pieces in the corpus (histogram bars) determines the grading schema we use for input models. The thick overlayed line shows the deduced normal curve.

Normal distribution tests

In our first corpus-based tests, we evaluate two single-value properties of a model: the *total duration* and *overall number of notes*.

To test similarity, we assume a normal distribution for both properties. We calculate the mean μ_c and standard deviation σ_c of all lengths are number of notes in the corpus. The normal distribution is scaled so the value of μ_c gives a fitness of 1. This results in the first two tests within our similarity test container. Figure 5 shows the deduced normal of the length test and the histogram of the lengths of the corpus, to show the correlation.

Descriptor correlation tests

In this system, a descriptor is the output of a series of transform methods applied to an input model. The rest of our tests look at similarity between the descriptors of the input and that of the corpus. Four transforms are applied to each of the three properties of a model, resulting in a total of twelve correlation tests. The input for these transforms is always one property of all notes, i.e. it is a row of the model matrix.

The following transforms are used to build the descriptor:

Histogram A histogram shows the distribution of the different values within the input data. For example, when analysing inter-onset intervals, a histogram represents the distribution of quarter notes and eighth notes etc. If two

[2] Downloaded from www.midiworld.com/bach.htm

descriptors have a similar histogram for a property, it means the distribution of that property is similar. The associated test discourages unlikely values (e.g. very low or very high pitches), where the likelihood of a value is determined by the corpus.

Histogram of differential This transform shows the distribution of the rate of change between consecutive notes. It represents the contour of a property. The associated test discourages unlikely changes (e.g. large rises or falls in pitch).

Fourier transform The previous transforms lose all time information, therefore they are not appropriate for testing repetition and structure. By applying a discrete Fourier transform to a property, we can see repetitive time-domain patterns appear as peaks in the spectrum. Similarity of these properties may encourage the emergence of rhythmic patterns and meter.

Fourier transform of differential This transform shows the repeating patterns in the rate of change between consecutive values.

All these transforms produce an output of size 128, independent of the size of the input. Therefore a descriptor is encoded as a matrix of size 12×128.

We calculate the correlation between two descriptors line-by-line, because we want to be able to assign different importances to the transforms. Therefore we produce 12 grades.

To calculate the correlation between the same line of two descriptors, we use the *Pearson correlation coefficient* (r). Given the ith lines from descriptors D and E, denoted by D_i and E_i, and their mean values denoted by $\bar{D_i}$ and $\bar{E_i}$, their correlation coefficient is given by the following equation:

$$r_{D_i,E_i} = \frac{\sum_{i=0}^{127}(D_i - \bar{D_i})(E_i - \bar{E_i})}{\sqrt{\sum_{i=0}^{127}(D_i - \bar{D_i})^2}\sqrt{\sum_{i=0}^{127}(E_i - \bar{E_i})^2}} \qquad (3)$$

The resulting coefficient is between -1 and 1. Positive values imply positive correlation, negative values imply negative correlation and a value of 0 implies no correlation. In our tests, we return 0 for negative values because we do not wish to obtain negative fitness scores. This approach allows us to return grades between the values of 0 and 1 without having to normalize the descriptors themselves.

Corpus clustering

We have defined correlation between two descriptors but our corpus consists of 30 different musical models, all of which have a different descriptor. Therefore a method must be used to incorporate the data from all 30 corpus members into the test. To verify how well this incorporation works, we evaluate the members of the corpus themselves using our tests.

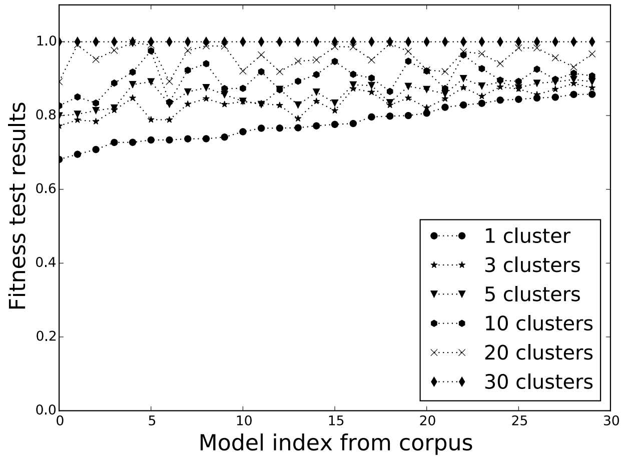


Figure 6: Similarity test results of corpus for different numbers of clusters; the models have been sorted by correlation.

A possible reference descriptor could be the centre of all descriptors from the corpus. This would be a *single niche* (Mahfoud, 1995) within our solution space. Experimenting with this approach reveals that testing the corpus itself gives relatively small grades (between 70 and 80%). This suggests that taking just one niche narrows down our solution space so much that even the corpus cannot achieve a high enough grade.

Another possible approach is to use all corpus members as niches, i.e. measure an input model's correlation to each of the corpus members and take the maximum value. In this case, all corpus members score 100%. However, this can become computationally expensive, and might broaden the solution space too much.

To obtain the best of both approaches, we *cluster* our corpus, partitioning the descriptors into K subsets. Afterwards we can use the centres of these clusters as reference descriptors, and test for the maximal correlation with a centre.

The clustering happens offline, before the evolutionary algorithm starts. It only needs to be done once therefore we do not have to worry about the computational expense. We use the *iterative partitional clustering algorithm* (Jain and Dubes, 1988), using the following steps:

1. Define the cluster centres as the first K descriptors.
2. Classify each descriptor into one of the K partitions by taking the minimal square error.
3. Find the new centres of gravity by taking the mean descriptor of each partition. Assign these as the new cluster centres.
4. Repeat steps 2-3 until no change occurs between iterations, or the number of steps reaches a maximum value (to avoid infinite oscillation).

This approach allows the flexibility of changing the value

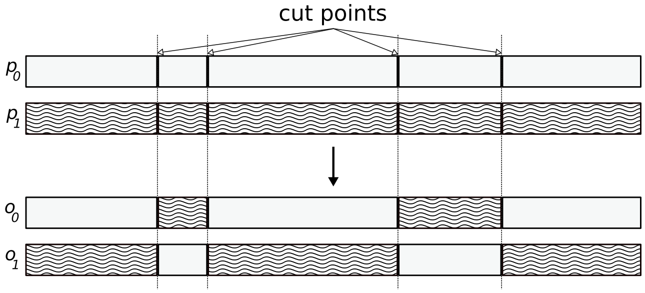


Figure 7: Visualization of crossover process; random cut points are chosen on the parents and offspring created from their recombination.

of K between runs. The two methods proposed initially can also be achieved by setting $K = 1$ or $K = 30$, respectively. Figure 6 shows the achieved grades of the corpus using different values for K.

Evolution strategy

Once all model grades have been calculated, we choose pairs of individuals for crossover. We use a *complementary phenotype selection* process (Dolin et al., 2002), which looks at individual test grades as well as overall scores. It chooses a mother based on roulette-wheel selection and builds hypothetical best-case offspring with all potential fathers. The hypothetical best-case offspring have the maximum grade for each test from the mother and father. The father is then chosen based on roulette-wheel selection applied to these grades.

When creating and breeding genetic strings, we do not concern ourselves with how this genetic string will be used later on, we simply view it as a byte array. In the first run of the algorithm, the generation-zero genetic strings are randomly generated byte arrays of the given size. In all subsequent generations, offspring are spliced versions of their parents. We will refer to the genetic strings of the parents as p_0 and p_1.

With the parents p_0 and p_1 chosen, the *genetic string breeder* builds two new genetic strings o_0 and o_1. It chooses a random number of cut points, separates p_0 and p_1 into chunks and populates the offspring (Figure 7). One chunk is taken from one parent, the next from the other.

Children o_0 and o_1 are then mutated by taking a random number of byte indices, and randomizing those bytes. The ratios of maximum cut points and maximum mutated bytes to the genetic string size are predefined for each run of the experiment.

Survival of the fittest is applied before the crossover process. A subset of the population can survive into the next generation. The ratio of the surviving units in a population is a predefined value. The units with maximum overall grades are allowed to survive into the next generation, but they are given an *age* value, so no phenotype can survive more than a given number of generations.

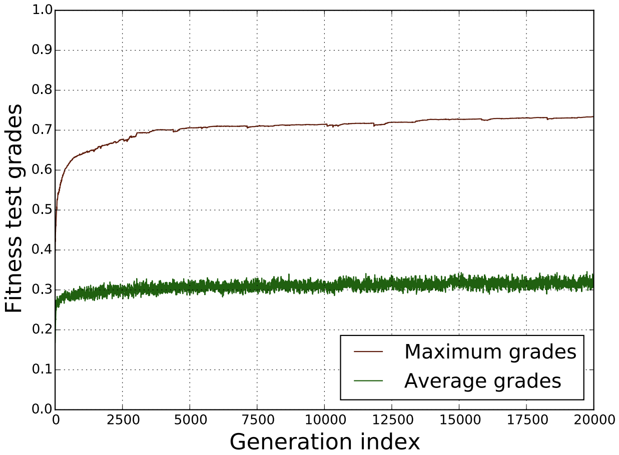


Figure 8: Progression of grades over 20000 generations: the mean and maximum grades for each generation, averaged over the 40 runs. The maximum grades steadily rise, while the mean grades seem to plateau quickly.

Experiments & results

For this paper we have run a total of 40 experiments, each time allowing the algorithm to reach 20000 generations; the parameters used here have been derived empirically from earlier test runs. We used a population size of 256 with a survival rate of 3% and number of clusters $K = 5$. The virtual machine's stopping conditions are a maximum of 60000 commands or 2500 output bytes. The breeder uses a 0.1% maximum cut point ratio and a 2% maximum mutation ratio.

In early experiments, it was found that some fitness tests tend to block others, preventing them from achieving high scores; importances have therefore been set to mitigate against this. For example, models with fewer notes have a higher probability of scoring well on correlation tests and this would block the emergence of desired lengths or number of notes. Therefore, the single property normal tests havea high importance.

The algorithm saves a number of files every 2500 generations. The first is the algorithm's current state, which can be imported for analysis or continuation. This file contains only the last generation. The second cyclically saved file holds all the test results since the past save. This way, none of them are lost, but only a limited number are in memory at any given time. The third file is a MIDI export of the highest-scoring model in the current generation.

To analyse the results of our series of tests, we take a look at the overall progression of the test results. Figure 8 shows the mean and maximum grades per generation, averaged over the 40 runs. We can observe a steady rise in

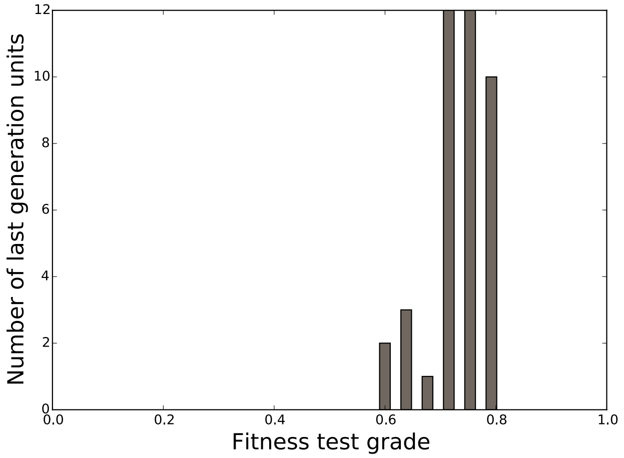


Figure 9: Distribution of highest-scoring individual grades. Every run managed to converge to at least a 60% correlation.

the maximum score, but stagnation in the mean values. This can be explained by the fragility of a genetic string when faced with crossover and mutation; even a small change can produce a completely different musical model.

Figure 9 shows the grade distribution of the highest-scoring individual in each run. It demonstrates that the algorithm gives consistent results on different iterations.

Examining the resulting MIDI files[3] allows us to evaluate, albeit subjectively, the musicality of the results. Their durations are universally within the boundaries dictated by the corpus statistics, although many achieve this with a smaller than desired number of notes (i.e. the notes are on average longer). Most exhibit recurring patterns of varying lengths, while some sound more random.

Figure 10 presents a selection from the results, demonstrating emergence of some musical properties, namely *repetition* and *variation*. Another property that emerges in rare cases is *segmentation*, i.e. longer sequences recurring at different times. Our tests did not specifically focus any of these properties, rather, they have emerged solely based on the similarity to the corpus.

Although repetition and variation appear in these pieces, other musical properties such as *harmony*, *melody* or *entropy*, are somewhat lacking. This suggests the need for further fitness tests inspired by music-theory.

Conclusion & future work

In this paper we have demonstrated a novel approach to evolutionary music composition: evolving the composition process rather than the product. By using a Turing-complete virtual machine, we have successfully defined our genotype as that process. Using a corpus of real music, we have derived tests to measure similarity of musical pieces to a set of niches.

Our results are promising, demonstrating that while this approach succeeds at converging towards the properties of the pieces in the corpus, is still only produces partially musical results. Some traits such as repetition and segmentation do arise, but there is still much room for improvement.

As a first step towards the reinforcement of the current ideas, we propose further test runs using different settings. The choice of parameters for the current run was largely empirical, therefore it may be interesting to assess differences arising from altering parameters such as population sizes, numbers of generations and survival rates.

For further research, we propose the inclusion of music-theory-inspired tests, which could accurately assess specific musical traits that do not yet emerge in the current context, such as harmony and entropy.

Improvements could also be made by a different choice of corpus. Since the current one had no musical expression, it would be interesting to explore what musical pieces played by human musicians could teach our system to do. Incorporation of velocity into the model properties would then allow us to test for accentuation and expressive tempo changes. We could also extend our corpus to include multivoice pieces such as string quartets. By testing the voices separately, the system could potentially detect the need for different pitch ranges or inter-onset intervals in the individual voices.

References

Biles, J. (1994). GenJam: A genetic algorithm for generating jazz solos. pages 131–137.

Biles, J. A. (2007). Improvizing with genetic algorithms: GenJam. In *Evolutionary Computer Music*. Springer London.

De Prisco, R., Zaccagnino, G., and Zaccagnino, R. (2011). A multi-objective differential evolution algorithm for 4-voice compositions. In *Differential Evolution (SDE), 2011 IEEE Symposium on*, pages 1–8.

Dolin, B., Arenas, M. G., and Guervós, J. J. M. (2002). Opposites attract: Complementary phenotype selection for crossover in genetic programming. In *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, PPSN VII, pages 142–152, London, UK, UK. Springer-Verlag.

Dostál, M. (2012). Musically meaningful fitness and mutation for autonomous evolution of rhythm accompaniment. *Soft Computing*, 16(12):2009–2026.

Eigenfeldt, A. (2009). The evolution of evolutionary software: Intelligent rhythm generation in kinetic engine. In *EvoWorkshops*, volume 5484, pages 498–507. Springer.

Eigenfeldt, A. (2012). Focus corpus-based recombinant composition using a genetic algorithm.

[3] All MIDI files and scores are uploaded to `http://csabasulyok.bitbucket.org/emc`

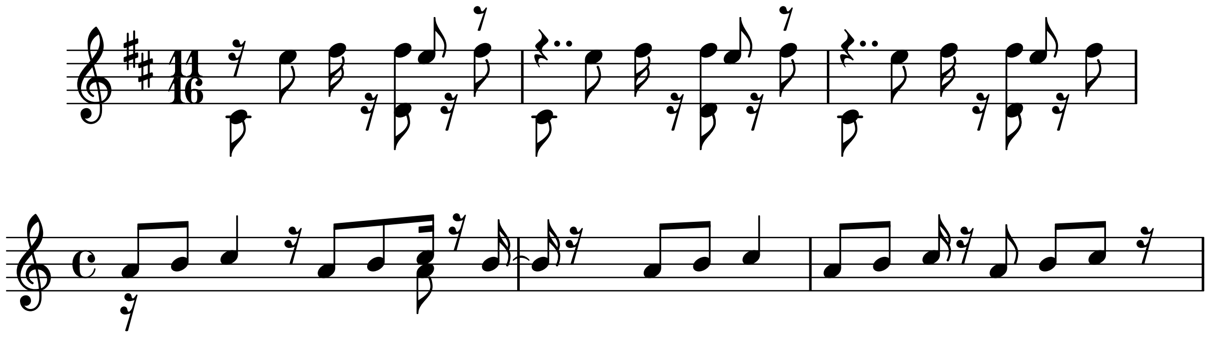


Figure 10: Two example segments from MIDI files created by the composition system. The recurring pattern in the top stave demonstrates that the algorithm helps repetitions emerge naturally. The lower stave shows a small three-note pattern with added decorations, demonstrating the evolution of variations. Time and key signatures have been added manually for illustration.

Fonseca, C. M. and Fleming, P. J. (1993). Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization.

Gibson, P. and Byrne, J. (1991). Neurogen, musical composition using genetic algorithms and cooperating neural networks. In *Artificial Neural Networks, 1991., Second International Conference on*, pages 309–313.

Hartmann, P. (1990). Natural selection of musical identities. In *International Computer Music Conference*.

Horner, A. and Goldberg, D. E. (1991). Genetic algorithms and computer-assisted music composition. In Belew, R. K. and Booker, L. B., editors, *ICGA*, pages 437–441. Morgan Kaufmann.

Jacob, B. (1995). Composing with genetic algorithms. In *International Computer Music Association*, pages 452–455.

Jain, A. K. and Dubes, R. C. (1988). *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

MacCallum, R. M., Mauch, M., Burt, A., and Leroi, A. M. (2012). Evolution of music by public choice. *Proceedings of the National Academy of Sciences*, 109(30):12081–12086.

Mahfoud, S. W. (1995). Niching methods for genetic algorithms. Technical report.

Manaris, B. Z., Roos, P., Machado, P., Krehbiel, D., Pellicoro, L., and Romero, J. (2007). A corpus-based hybrid approach to music analysis and composition. In *AAAI*, pages 839–845. AAAI Press.

Martins, J. M. and Miranda, E. R. (2007). Emergent rhythmic phrases in an a-life environment.

McIntyre, R. (1994). Bach in a box: the evolution of four part baroque harmony using the genetic algorithm. In *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, pages 852–857 vol.2.

Miranda, E. R. and Biles, J. A. (2007). *Evolutionary Computer Music*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Phon-amnuaisuk, S., Tuson, A., and Wiggins, G. (1999). Evolving musical harmonisation. In *In ICANNCA*.

Rodriguez, J. D. F. and Vico, F. J. (2014). AI methods in algorithmic composition: A comprehensive survey. *CoRR*, abs/1402.0585.

Tokui, N. and Iba, H. (2000). Music composition with interactive evolutionary computation. In *Proceedings of the 3rd international conference on generative art*, volume 17, pages 215–226.

Towsey, M., Brown, A., Wright, S., and Diederich, J. (2001). Towards melodic extension using genetic algorithms. *Educational Technology & Society*, 4(2).

Unemi, T. (2002). SBEAT3: A tool for multi-part music composition by simulated breeding.

Waschka, R. I. (2007). Composing with genetic algorithms: GenDash. In *Evolutionary Computer Music*. Springer London.

Whorley, R. P., Rhodes, C., Wiggins, G., and Pearce, M. T. (2013). Harmonising melodies: Why do we add the bass line first? In *International Conference on Computational Creativity*, pages 79–86, Sydney.

Does self-replication imply evolvability?

Thomas LaBar[1,2], Christoph Adami[1,2] and Arend Hintze[2,3,4]

[1]Department of Microbiology and Molecular Genetics
[2]BEACON Center for the Study of Evolution in Action
[3]Department of Integrative Biology
[4]Department of Computer Science and Engineering
Michigan State University, East Lansing, MI 48824
hintze@msu.edu

Abstract

The most prominent property of life on Earth is its ability to evolve. It is often taken for granted that self-replication–the characteristic that makes life possible–implies evolvability, but many examples such as the lack of evolvability in computer viruses seem to challenge this view. Is evolvability itself a property that needs to evolve, or is it automatically present within any chemistry that supports sequences that can evolve in principle? Here, we study evolvability in the digital life system Avida, where self-replicating sequences written by hand are used to seed evolutionary experiments. We use 170 self-replicators that we found in a search through 3 billion randomly generated sequences (at three different sequence lengths) to study the evolvability of *generic* rather than hand-designed self-replicators. We find that most can evolve but some are evolutionarily sterile. From this limited data set we are led to conclude that evolvability is a likely–but not a guaranteed– property of random replicators in a digital chemistry.

Introduction

For life of the type as we experience it on Earth to emerge from an initially abiotic state requires two seemingly independent things to happen. First, a self-replicator has to emerge (or else, to arrive on Earth from extraterrestrial sources (Arrhenius, 1908; Hoyle and Wickramasinghe, 1981; Wickramasinghe, 2011)). Secondly, this self-replicator must be able to evolve and thus diversify into the complexity we see today. In general we think of self-replicators as extremely rare (von Neumann, 1966). But in order to jump-start the process of evolution, these rare self-replicators now also must be endowed with another property–evolvability (Wagner and Altenberg, 1996; Kirschner and Gerhart, 1998; Wagner, 2005). By evolvability here we mean the ability to produce variants or mutants (produced by a faulty self replication process or else by external noise) that can also self-replicate. Without evolvability the self-replicator would just multiply but not adapt. While it is safe to assume that one of the first self-replicators on Earth was evolvable ($n = 1$), it is not at all clear whether evolvability is a general property of self-replicators, or else is an additional constraint that renders the emergence of life even more improbable. Here we use the computational evolution system Avida (Adami and Brown, 1994; Adami, 1998; Ofria et al., 2009) to investigate whether random self-replicators, that is, randomly generated sequences of code written in the avidian instruction set that happen to be able to self-replicate, also are automatically endowed with the capacity to evolve. Even though evolvability appears to be inherent to avidians, we cannot rule out a priori whether the observed evolvability of avidians is a consequence of the evolvabilty of the hand-written ancestor or is instead a germane property of all self-replicators that can exist within this digital chemistry. To resolve this question, we searched for self-replicating sequences that we generated randomly, within the three size classes $L = 8$, $L = 15$ (the size of the standard avidian self-replicator), and $L = 30$. We found 170 such sequences after generating 3 billion random sequences, and report their evolvabilty below.

Self-replicators in Avida

In Avida, small self-replicating computer programs ("avidians") compete for limited memory space and limited CPU resources needed to successfully self-replicate (see Ofria et al. 2009 for a complete description of the Avida system). This ability to self-replicate is contained within an individual avidian's genome of instructions (see Fig. 1). Because this genome is then passed on to an avidian's descendants, the ability to self-replicate is heritable. Selection in Avida is implicit, since faster replicators have a higher chance of making offspring. During the process of self-replication, mutations may be introduced, resulting in error-prone replication and variation within the population. Thus, Avida is an *instance* of evolution by natural selection (Pennock, 2007), because it contains inheritance, variation, and differential fitness among individuals. Avida has been used to explore a diverse set of topics in evolutionary biology such as the evolution of genomic complexity (Lenski et al., 1999), the "survival of the flattest" effect at high mutation rates (Wilke et al., 2001), the evolution of complex features (Lenski et al., 2003), the evolution of reproductive division of labor (Goldsby et al., 2014), and the role of coevolution in the origin of complexity (Zaman et al., 2014).

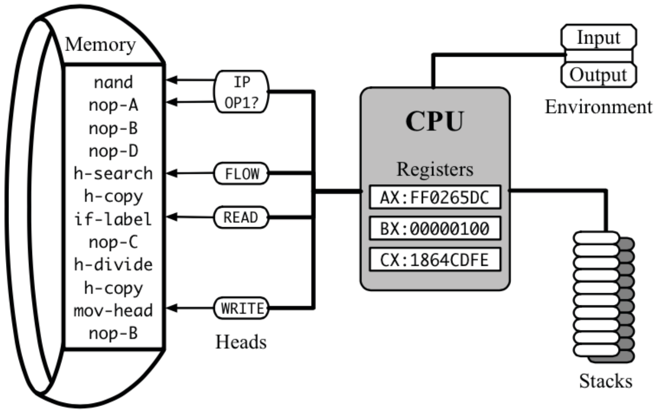


Figure 1: The avidian CPU in the process of executing a segment of code. The CPU uses three registers (AX,BX,CX) as well as an instruction pointer (IP) that reads the program into the CPU. A read-head, a write-head, and a flow-head are used to specify positions in the CPU's memory. The 'copy' command reads from the read-head and writes to the write-head, while 'jump'-type statements move the instruction pointer to the flow-head. The CPU uses two stacks to simulate an "infinite Turing tape", while input/output buffers serve to communicate between the CPU and its environment (reproduced from Ofria et al. (2009), with permission).

Avidian genomes are composed of usually 26 different instructions, typically rendered as a string of lowercase letters, where each letter corresponds to one command (see Table 1). These instructions can be considered analogous to the 20 amino acids common to all biological organisms.

A self-replicator in Avida must have a set of instructions in the right order to: first allocate memory, then copy itself into this newly allocated memory, and finally it has to split the new memory from the old one in order to crate a new organism. These instructions and their interactions are complex and in general it is not possible to predict if an organism will replicate just by examining the sequence. As a consequence, the ability to self-replicate has to be tested directly by allowing the instructions to execute (we surmise that this problem of predicting the ability to self-replicate is similar to the Halting Problem (Turing, 1936), where it is necessary to run the algorithm to find out if it halts or not).

In the majority of all Avida experiments a default hand-designed "start" organism is used to seed a population. This choice is to some extent historical: the designers assumed that self-replicators were too rare to be found by a random process (Adami, 1998). Indeed, any particular sequence of length L=15 for example (the length of one of the standard hand-written ancestors) has a likelihood of $p = 26^{-15} \approx 6 \times 10^{-22}$. If a million of such sequences could be checked per second, on one thousand CPUs running in parallel, it would take about 50,000 years to find it. As a remedy, the designers wrote one instead (most of the common ancestors in Avida were written by Charles Ofria). However, it is clear that the density of self-replicators in the space of all sequences depends on the chemistry (here, the instruction set) used. Since the inception of Avida in 1993, the standard instruction set has changed, and it appears that using the current set (that is, the current "chemistry") self-replicators can be found among randomly generated sequences (as reported here) because the information content of the sequences is significantly smaller than 15.

No self-replicator in nature (or within Avida) replicates without error, because noise in the system is inevitable and will always affect replication by increasing variation. Variation in Avida can have two different causes. First, it is possible that the sequence of commands performs in such ways that the resulting copy is modified. This typically leads to repeated commands, insertions, or deletions. Such changes are inherent to the self-replicator (they are genetically encoded and thus deterministic) and referred to as "implicit" mutations. A second mechanism is less deterministic, and occurs during the processes of division and reproduction, processes that are inherently made to be error-prone to a degree that can be specified by the user. These copy-errors, as well as the probabilistic insertion and deletion of instructions or code snippets reflect the noisiness of the system and are not under control of the organism itself. Note that even the most sophisticated polymerases that also have proof reading abilities cannot create perfect replicates every time, while this is in principle possible for avidians without implicit mutations by turning the mutation rate to zero. While biochemical-based life does not have a mutational mechanism similar to the implicit mutations seen in avidians, it is not unreasonable to assume that early replicators also underwent large mutations due to a lack of error-correction during replication.

Evolvability in Avida

Evolvability describes the ability of an organism to undergo evolutionary adaptation (Wagner and Altenberg, 1996; Kirschner and Gerhart, 1998; Wagner, 2005). There are two main pathways to adaptation in Avida. The first is *optimization*, where organisms with a minimal replication time outcompete others in the population (akin to r-selection (Pianka, 1970)). The second is *innovation*, where avidians can evolve new phenotypic traits that enable a fitness increase (the K-selection mode). These phenotypic traits are the ability to perform certain Boolean logic operations; these logic operations allow an individual to execute a greater proportion of its genome than other avidians that do not perform such logic operations. This indirectly allows them to self-

Instruction	Description	Symbol
nop-A	no operation (type A)	a
nop-B	no operation (type B)	b
nop-C	no operation (type C)	c
if-n-equ	Execute next instruction only-if ?BX? does not equal complement	d
if-less	Execute next instruction only if ?BX? is less than its complement	e
if-label	Execute next instruction only if template complement was just copied	f
mov-head	Move instruction pointer to same position as flow-head	g
jmp-head	Move instruction pointer by fixed amount found in register CX	h
get-head	Write position of instruction pointer into register CX	i
set-flow	Move the flow-head to the memory position specified by ?CX?	j
shift-r	Shift all the bits in ?BX? one to the right	k
shift-l	Shift all the bits in ?BX? one to the left	l
inc	Increment ?BX?	m
dec	Decrement ?BX?	n
push	Copy value of ?BX? onto top of current stack	o
pop	Remove number from current stack and place in ?BX?	p
swap-stk	Toggle the active stack	q
swap	Swap the contents of ?BX? with its complement	r
add	Calculate sum of BX and CX; put result in ?BX?	s
sub	Calculate BX minus CX; put result in ?BX?	t
nand	Perform bitwise NAND on BX and CX; put result in ?BX?	u
h-copy	Copy instruction from read-head to write-head and advance both	v
h-alloc	Allocate memory for offspring	w
h-divide	Divide off an offspring located between read-head and write-head	x
IO	Output value ?BX? and replace with new input	y
h-search	Find complement template and place flow-head after it	z

Table 1: Instruction set of the avidian programming language used in this study. The notation ?BX? implies that the command operates on a register specified by the subsequent nop instruction (for example, nop-A specifies the AX register, and so forth). If no nop instruction follows, use the register BX as a default. More details about this instruction set can be found in Ofria et al. (2009).

replicate at a greater rate (see, for example, Adami 1998, 2006). These different modes of survival can be thought of as different niches that avidians can inhabit (White and Adami, 2004).

It is possible in general that an organism (digital or otherwise) carries such a specific sequence of nucleotides or (as in Avida) commands that every possible mutation prevents self-replication. From a fitness landscape point of view, such replicators would represent an isolated fitness peak where self replication is only possible at the top, and where each mutation is lethal. Such self-replicators, born at the top of the fitness peak, so to speak, would be unevolvable. A self-replicator that can evolve, on the other hand, would never emerge at peak in the landscape, but rather on a lower level and subsequently find positive and or neutral mutations that ultimately lead to higher fitness levels in the landscape. We know that the hand-written replicators in Avida are evolvable, but it is not clear how likely evolvability is among randomly generated replicators.

Methods

All experiments are performed using Avida version 2.14, which can be downloaded from https://github.com/devosoft/avida. We first randomly generated Avida sequences to discover individuals that could self-replicate, using a uniform random distribution for each command at each site, which ensures that each sequence has the same likelihood to be generated, given by

$$p = \frac{1}{D^L}\ , \tag{1}$$

where D is the size of the alphabet (D=26 here) and L is the length of the sequence generated (Adami and LaBar, 2015). To decide whether a sequence could successfully self-replicate, it must pass two tests. First, we tested whether the organism could successfully divide within its lifespan. Here, we used standard Avida parameters for an organism's lifespan: it must divide before it executes $20 \times L$ instructions. This test indicates that an avidian can successfully reproduce, it does not imply that its descendants also can reproduce. In our search we discovered many viable avidians that were able to successfully divide into two non-viable organisms. Therefore, we only counted sequences that could replicate and produce offspring that could also replicate as true self-replicators (in other words, they are "colony-forming"). This does not imply that every self-replicator produces a perfect copy of itself in the absence of mutation. Indeed, most of these replicators undergo implicit mutations solely due to their genome sequence, and their offspring differ in length from the parent. In analyzing a genome's ability to self-replicate, we used the default Avida settings, described for example in (Ofria et al., 2009). For our study of evolvability, we also included the default length 15 hand-written ancestor in our set of self-replicators.

In order to test whether these self-replicators could optimize their fitness, we evolved them in an environment where *only* decreased self-replication speed was under positive selection (the r-selection niche). We evolved these replicators for 10^3 generations at a population size of $3,600$ individuals (10 replicates). Instruction mutations occurred at a genomic rate of 0.1 at division (meaning the likelihood to incur an error is length-independent), and both insertions and deletions occurred at a genomic rate of 0.005 per division. We measured an organism's evolvability as its gain in relative fitness at the end of the experiment.

Finally, we tested the self-replicators' ability to evolve greater complexity by evolving new phenotypic traits (that is, in the K-selection mode). The traits that are rewarded are logical functions that an avidian can solve by stringing together code involving the `nand` function (given by the letter `u`, see Table 1). In this setting, nine different logic tasks may be rewarded, denoted as the bit-wise `NOT, NAND, AND, OR-NOT,OR, AND-NOT, NOR, XOR,` and `EQU`. We evolved the self-replicators of length 15 in an environment where nine Boolean logic operations, and hence nine phenotypic traits, are under positive selection; this is often referred to as the logic-9 environment (Lenski et al., 2003). For this experiment, we evolved these replicators for 10^4 generations at a population size of 10^4 individual (ten replicates). Increased population size and experiment time was used to allow time for trait evolution. The mutation rates were the same as in the optimization experiments. We quantified an organism's evolvability in these experiments by measuring the number of evolved phenotypic traits.

Results

Of the 10^9 randomly-generated Avidian genomes in each length class, we found 6 self-replicators of length 8, 58 self-replicators of length 15, and 106 self-replicators of length 30 (see Table for their sequences). However, it is unlikely that each self-replicator needs every instruction in its genome in order to self-replicate (Adami, 2015). Many of these self-replicators could be unstable and would generate implicit mutations upon replication. Therefore, we tested the replication fidelity of each self-replicator (Fig. 2). All replicators of length 8 were able to undergo error-free replication without external noise (mutation rate set to zero). This is likely due to the fact that avidian genomes cannot be shorter than length 8; however, that does not prevent the occurrence of genomes that replicate to a length larger than 8, which we did not see among this set. Self-replicators of length 15 and 30 could not replicate error-free, even without external noise, and most of their genome sequences decreased upon successful replication.

Next, we tested the evolvability of these replicators in the sense of how well they could optimize their genomes as a way to increase replication speed, and thus their fitness. We found that the majority of these spontaneous self-replicators

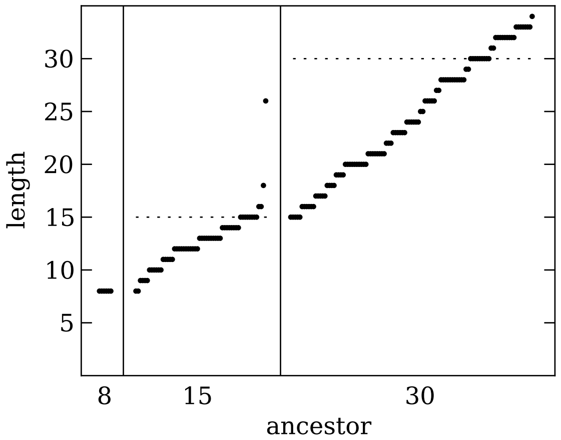


Figure 2: Change in length after the first round of replication for all self replicators we found of length 8, 15, and 30. The self-replicators are ordered by length increase within their individual groups. The dashed lines indicate the start organism's length.

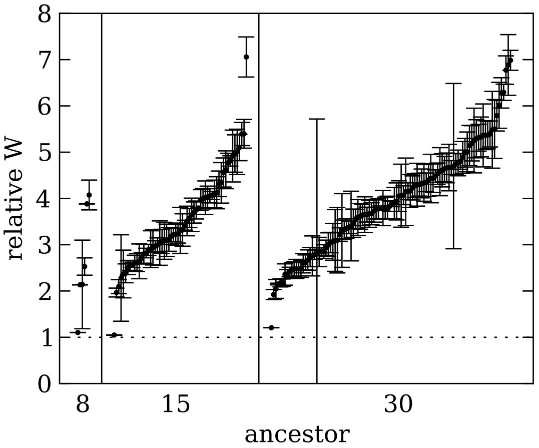


Figure 3: Relative fitness after 1,000 generatios of evolution for all replicators we found of length 8, 15, and 30. The replicators are ordered by relative fitness increase within their groups of similar length. The fitness of all start organisms is normalized to 1, indicated by the dashed line. The error bars indicate the standard deviation over 10 replicate evolution trials with the same ancestor.

of length 8, 15, and 30 possess the ability to optimize their replication algorithm (see Figure 3). While the increase in fitness across the different self-replicators varied due to differing ancestral fitness, most increased in fitness by more than two-fold over the course of 10^3 generations of evolution. However, we found a few replicators (3%) that were evolutionary sterile (defined as a relative fitness < 2), demonstrating that not all self-replicators can easily undergo significant adaptation.

Because the ability to optimize replication speed does not automatically imply the ability to evolve greater complexity, we also tested at the ability of the length 15 replicators to evolve new phenotypic traits (see Methods for a description of those traits). All 58 self-replicators were able to evolve at least one phenotypic trait in one replicate, although the likelihood of the emergence of phenotypic traits varied greatly across the different replicators and within the replicates for each specific replicator (see Fig. 4). Moreover, most self-replicators were able to evolve multiple traits. The hand-written Avida ancestor of length 15 displayed the *least* evolvability in regards to trait evolution compared to the 58 randomly-generated self-replicators. Only one out of ten replicates evolved any phenotypic traits in the allotted time, although that replicate did manage the evolution of two traits.

Discussion

Here, we asked if spontaneous self-replicators in Avida would all possess the additional ability to change in such a way that they could initiate evolution. We showed that the majority of self-replicators are robust enough to tolerate mutations as well as changes to their genome size. This suggests that self-replicators in this digital chemistry are robust and thus will most likely be able to jump-start evolution. Further, this result holds both when we looked at the ability of self-replicators to increase fitness through optimization or through the evolution of new phenotypes. In addition, many spontaneous self-replicators made faulty copies of themselves in the absence of external noise, which further supports the idea that variability is an inherent property of spontaneous self-replication.

These results confirm that not every spontaneous self-replicator necessarily is evolvable. On the other hand, we also find that the majority of spontaneous self-replicators deterministically do not make exact copies even in the absence of mutation, questioning the term "self-replicator" for these genotypes. Indeed, while these spontaneously generated sequences are "colony-forming" (in the sense that they produce sequences that can also replicate themselves), these are not self-replicators in the strictest sense of the definition. Instead, these replicators mutate themselves to become a self-replicator: we call these individuals "proto-self-replicators" or "proto-replicators". This means that the number of potential sequences that can start evolution is increased by existence of these proto-replicators. In natural chemistries, this phenomenon has been discussed widely (Bernal, 1949; Miller et al., 1953; Eigen, 1971; Miller and Orgel, 1974; Eschenmoser and Loewenthal, 1992), suggesting that the first self-replicator might have needed either self organisation or some other forms of catalysis or autocatalysis in order to

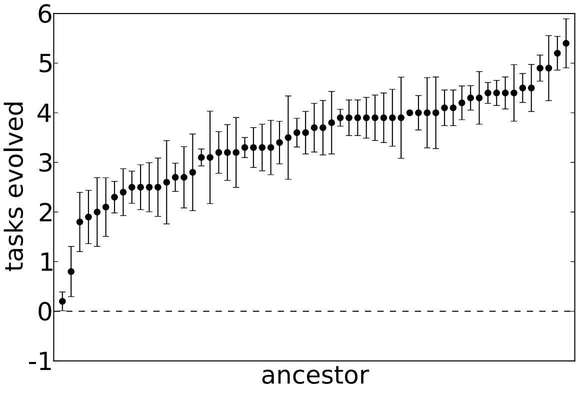


Figure 4: The number of logic tasks that spontaneous replicators of length 15 were able to perform after $10,000$ generations of evolution in the Avida logic-9 environment. Replicators are ordered by number of tasks achieved. Error bars indicate the standard error of the mean for 10 replicates per ancestral organism.

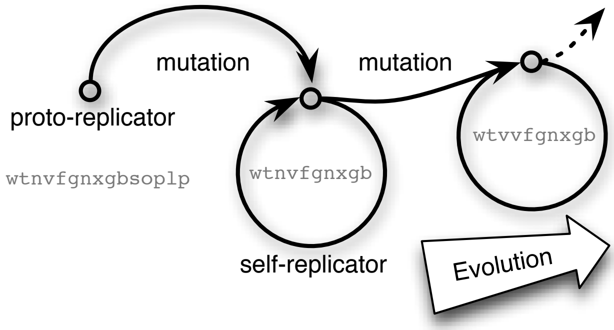


Figure 5: Illustration how an Avida proto-replicator (left) "copies" itself into a self-replicator (middle). This replicator can now experience external mutations, which leads to a new self replicator. The repetition of such process is what jump-starts evolution (right). The text string is a program example from Avida.

create the right chemistry to start polymerization in the first place. Figure 5 gives an illustration of this concept.

Leo Tolstoy famously begun his "Anna Karenina" by remarking that *"Happy families are all alike; every unhappy family is unhappy in its own way."*. In the same manner we asked here whether, with respect to evolvability, all self-replicators are alike (while of course all non-replicators are non-replicating in their own way). Our results show that not every self-replicator is suitable as the ancestor for evolution. But we also find that, with respect to the capacity to evolve functional complexity, random replicators are better than even those designed by thinking humans.

Our results further suggest that life, if possibly found elsewhere, does not necessarily experience evolution. We might at some point in the future find self-replicators that maintained their ancestral form due to their inability to tolerate enough variation to evolve. Similarly, it seems possible that, if such an mutationally-fragile self-replicator exists, it might be outcompeted by other self replicators that can take advantage of mutations, and thus attain higher fitness and ultimately outcompete those mutationally-fragile self-replicators. Thus, as long as evolvability is evolvable, complexity should ensue even when sparked by the most humble and awkward replicators.

References

Adami, C. (1998). *Introduction to Artificial Life*. Springer Verlag, New York and Heidelberg.

Adami, C. (2006). Digital genetics: Unravelling the genetic basis of evolution. *Nat Rev Genet*, 7:109–118.

Adami, C. (2015). Information-theoretic considerations concerning the origin of life. *Origins of Life and Evolution of Biospheres*, 45.

Adami, C. and Brown, C. (1994). Evolutionary learning in the 2D Artificial Life system Avida. In Brooks, R. and Maes, P., editors, *Proc. Artificial Life 4*, pages 377–381. MIT Press.

Adami, C. and LaBar, T. (2015). From entropy to information: Biased typewriters and the origin of life. In Walker, S., Davies, P., and Ellis, G., editors, *Information and Causality: From Matter to Life*. Cambridge University Press, Cambridge, MA.

Arrhenius, S. (1908). *Worlds in the Making: The Evolution of the Universe*. Harper and Row, New York, NY.

Bernal, J. (1949). The physical basis of life. *Proceedings of the Physical Society. Section B*, 62:597.

Eigen, M. (1971). Selforganization of matter and the evolution of biological macromolecules. *Naturwissenschaften*, 58:465–523.

Eschenmoser, A. and Loewenthal, E. (1992). Chemistry of potentially prebiological natural products. *Chemical Society Reviews*, 21:1–16.

Goldsby, H. J., Knoester, D. B., Ofria, C., and Kerr, B. (2014). The evolutionary origin of somatic cells under the dirty work hypothesis. *PLoS biology*, 12:e1001858.

Hoyle, F. and Wickramasinghe, N. (1981). *Evolution from Space*. Simon & Schuster Inc, New York, NY.

Kirschner, M. and Gerhart, J. (1998). Evolvability. *Proc. Natl. Acad. Sci. U.S.A.*, 95:8420–8427.

Lenski, R. E., Ofria, C., Collier, T. C., and Adami, C. (1999). Genome complexity, robustness and genetic interactions in digital organisms. *Nature*, 400:661–664.

Lenski, R. E., Ofria, C., Pennock, R. T., and Adami, C. (2003). The evolutionary origin of complex features. *Nature*, 423:139–144.

Miller, S. L. et al. (1953). A production of amino acids under possible primitive earth conditions. *Science*, 117:528–529.

Miller, S. L. and Orgel, L. E. (1974). *The origins of life on the earth*. Prentice-Hall Englewood Cliffs, NJ.

Ofria, C., Bryson, D., and Wilke, C. (2009). Avida: A software platform for research in computational evolutionary biology. In Adamatzky, A. and Komosinski, M., editors, *Artificial Life Models in Software*, pages 3–35. Springer Verlag, 2nd edition.

Pennock, R. T. (2007). Models, simulations, instantiations and evidence: The case of digital evolution. *Journal of Experimental and Theoretical Artificial Intelligence*, 19:29–42.

Pianka, E. (1970). On r and K selection. *American Naturalist*, 104:592–597.

Turing, A. M. (1937). On computable numbers, with an application to the Entscheidungsproblem. *Proc. London Math. Soc. Ser. 2*, 42:230-265.

von Neumann, J. (1966). *The Theory of Self-reproducing Automata*. edited and completed by A. Burks. Univ. of Illinois Press, Urbana, IL.

Wagner, A. (2005). *Robustness and Evolvability in Living systems*. Princeton University Press, Princeton, NJ.

Wagner, G. P. and Altenberg, L. (1996). Complex adaptations and the evolution of evolvability. *Evolution*, 50:967–976.

White, J. S. and Adami, C. (2004). Bifurcation into functional niches in adaptation. *Artif Life*, 10:135–44.

Wickramasinghe, C. (2011). Bacterial morphologies supporting cometary panspermia: A reappraisal. *International Journal of Astrobiology*, 10:25–30.

Wilke, C. O., Wang, J. L., Ofria, C., Lenski, R. E., and Adami, C. (2001). Evolution of digital organisms at high mutation rates leads to survival of the flattest. *Nature*, 412:331–333.

Zaman, L., Meyer, J. R., Devangam, S., Bryson, D. M., Lenski, R. E., and Ofria, C. (2014). Coevolution drives the emergence of complex traits and promotes evolvability. *PLoS Biology*, 12:e1002023.

Appendix: Sequences of replicators

Table 2: Sequences of 8-mer replicators and 15-mer replicators. The sequence denoted (1) is the hand-written L=15 ancestor. The 106 proto- and self-replicators of length L=30 can be downloaded from http://dx.doi.org/10.6084/m9.figshare.1416247

$L = 8$			
qxrchcwv	vxfgwjgb	wxvxfggb	vhfgxwgb
wxrchcvz	wvfgjxgb		
$L = 15$			
wtnvfgnxgbsoplp	mwtepvfgskvrxgb	wvxvfagbchcmwse	nvlvfgqnsxwgbzf
wlmvmfgowxdogbf	wvfgmwxgbjqpylp	vovvwnfgxlrgbjn	prvfgemsxwepgbo
vfgxqhmwmfjphgb	vtwfgxgbyhdnahk	vlfgvuiofxwgbpc	dywqvphfguxqdgb
wvufguuxltsgbwd	vmfifgwvpowxgbt	wzvfgqxrpoujgbn	jivfgzwkjxogbtw
wvfgnxwhgbaplye	vwowfgqxqwxgbjo	vvfgwxqwdfogbhq	wvfgofeoxxgbrmg
wvfguzxqjgbiokn	vkhwfgyfrxwgbtr	vfgwvtljvrwxgbl	vfglqwxljgbwdsf
lwvfgxoetfdgbhp	lsvrwfgxmmwgbwg	vnfgudsftwxwhgb	vlfgvmhwuxwlgbq
rrowvmfgxjgbuyx	drvfgwioxrmgbjx	inoidjwvfglxgbd	iwlkvfgxgbsslez
vfgwpxpgbyxxddi	nvqwqfgfnoxpgbm	vfgkqldxidwgbxt	vnwdfgxgbyeevbg
vmfgqwrmdkyxuhc	qvufgxwdgbojyom	vhfgtxlwgbfbryb	wrvvnfgxmgbctol
vwqqfgmxgbeixsh	vqvfgvqxwygbwzn	vyhfgxwkypgbyny	yvdwfgxgbvwrfpg
wvfghqtzoxjirgb	irwovfgxjgbwhbr	wvfgxdmoprllwgb	liwvlfguxgbnhsn
vfgfyxnxwmgbtmk	vfgxqdrkswgbpgz	vfgwpmmxhqgbkmc	svkfgxlujlwmgbx
vwtsrlfgxhgbije	vpdtwfgkfkxgbkp	zvyfgsdwxjgbzdn	kvdsovwfgxgbyhf
vmnfgmxxwigbcc	qwvqfgfwxxfegbg	wzcagczvfcaxgab$^{(1)}$	

Effects of Social Network Size and Topology on Evolutionary Decision Making

Hiroki Sayama Shelley D. Dionne Francis J. Yammarino

Binghamton University, State University of New York, USA
sayama@binghamton.edu

Collective decision making is crucial in human organizations and societies. When a collective is working on exploration of problem space and/or ideation for creative solutions, the evolutionary perspective is useful for conceptualizing and modeling collective decision making, where populations of ideas spread and evolve on a social network habitat via continual applications of evolutionary operators by human individuals (Sayama & Dionne 2015).

Using an evolutionary approach to model collective decision making, we conducted agent-based simulations to investigate how collective decision making would be affected by the size and topology of social network structure (Sayama, Dionne & Yammarino 2013). In our model, each agent has its own utility function defined over a multi-dimensional problem space, which is marginally different from the "true" utility function that is not accessible from any agent. Each agent can memorize multiple ideas in mind, and iteratively applies evolutionary operators (e.g., replication, mutation, recombination, elimination) to the idea population it has. The outcomes of evolutionary operations are stored in the agent's mind, and also sent to the neighbors to which the agent is connected. This allows the spread of ideas through social ties.

Each simulation was run for a fixed number of iterations, and then the level of idea convergence at a social level and the true utility value of the most supported idea were measured as the performance metrics of collective decision making. The former metric characterizes the ability for the society to form consensus, while the latter characterizes its ability to find the true best solution. The size of the network (number of agents or nodes) was varied from 5 to 640 logarithmically. Three network topologies with the same average node degree were tested: Random, small-world and scale-free. For more details of the model and the simulation settings, see Sayama, Dionne & Yammarino (2013).

Simulation results indicated that expanding the size of the social network generally improved the quality of ideas at the cost of decision convergence. Simulations with different social network topologies further revealed that collective decision making on small-world networks with high local clustering tended to achieve highest decision quality more often than on random or scale-free networks (Fig. 1). This can be understood in that local clustering helps agents in different regions in a network maintain their respective focus areas and engage in different local search, possibly enhancing the effective parallelism of collective decision making and therefore resulting in a greater number of successful decisions. In contrast, the links in random and scale-free networks are all "global", mixing discussions prematurely and therefore reducing the effective parallelism of collective decision making. These observations have an interesting contrast with the fact that random and scale-free networks are highly efficient in information dissemination because of their global connectedness. Our results indicate that such efficiency of information dissemination may not necessarily imply the same for collective decision making. This work may also offer an evolutionary explanation for the nontrivial relationship between network clustering and problem solving, a problem that is actively debated in organizational science (Shore, Bernstein & Lazer 2014).

References

Sayama, H., Dionne, S. D., & Yammarino, F. J. (2013). Evolutionary perspectives on collective decision making: Studying the implications of diversity and social network structure with agent-based simulations. http://arxiv.org/abs/1311.3674

Sayama, H. & Dionne, S. D. (2015). Studying collective human decision making and creativity with evolutionary computation. *Artificial Life*, in press.

Shore, J., Bernstein, E., & Lazer, D. (2014). Facts and figuring: An experimental investigation of network structure and performance in information and solution spaces. *Harvard Business School Organizational Behavior Unit Working Paper*, 14-075.

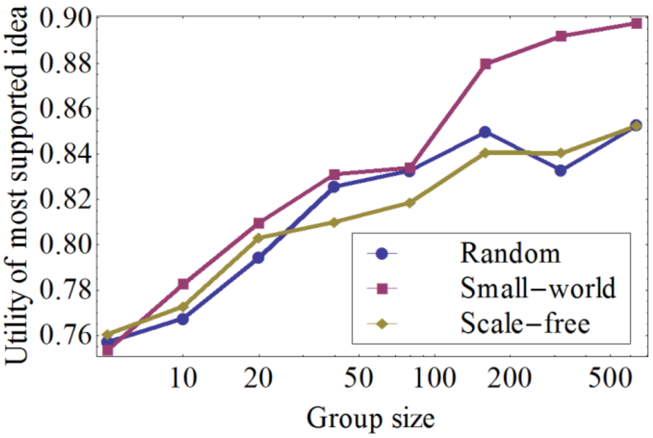


Figure 1. Effects of size and topology of social networks on the true utility value of the collective decision (most supported idea) made through idea evolution on the networks of simulated agents. Each dot represents an average result of 500 independent simulation runs. For larger networks (size > 100), the small-world topology outperformed the other two topologies in terms of the quality of the collective decision.

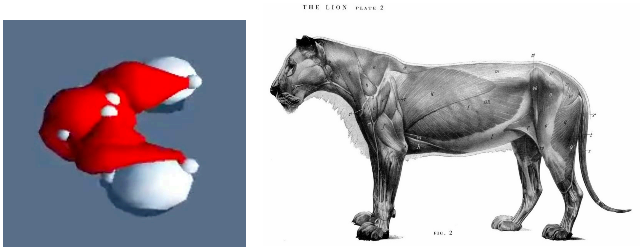
Soft-Body Muscles for Evolved Virtual Creatures: The Next Step on a Bio-Mimetic Path to Meaningful Morphological Complexity

Dan Lessin   Sebastian Risi

IT University of Copenhagen
Copenhagen, Denmark
dles@itu.dk, sebr@itu.dk

Abstract

In the past, evolved virtual creatures (EVCs) have been developed with rigid, articulated bodies, and with soft bodies, but never before with a combination of the two. In nature, however, creatures combining a rigid skeleton and non-rigid muscles are some of the most complex and successful examples of life on earth. Now, for the first time, creatures with fully evolved rigid-body skeletons and soft-body muscles can be developed in the virtual world, as well. By exploiting and re-purposing the capabilities of existing soft-body simulation systems, we can evolve complex and effective simulated muscles, able to drive a rigid-body skeleton. In this way, we can begin to bridge the gap between articulated and soft-bodied EVCs, and take the next step on a nature-inspired path to meaningful morphological complexity for evolved virtual creatures.



(a)   (b)

Figure 1: The inspiration for this paper's approach (a) to morphological complexity is real world creatures (b) in which a rigid skeleton is driven by muscles. See results in motion at `http://goo.gl/rvSvFv`.

Introduction

Since evolved virtual creatures (EVCs) were first introduced (Sims, 1994b), the standard implementation has employed a rigid-segmented skeleton-like body, with only minor variations. There have been some recent investigations into rigid-segmented bodies with more complex segment forms (Auerbach and Bongard, 2012), and soft-bodied creatures have also produced compelling results (Rieffel, 2013; Cheney et al., 2013), but no attempt has yet been made to combine the two approaches. Evolution in the natural world has employed such a combination to produce a great variety of highly complex and successful creatures (Figure 1b). What might emerge when this method of embodiment is reproduced from life-as-we-know-it in the real world to life-as-it-could-be in the virtual world?

To begin this pursuit, appropriate simulated muscles are required. Much of the previous related work has applied simulated muscles to the more limited case of fixed-morphology creatures. These include inflated-cloth muscles on simulated robots (Glette and Hovin, 2010), muscle-inspired joint drives for simulated swimming robots (Moore and McKinley, 2014), and relatively complex spring-like muscles used to animate human-designed morphologies for entertainment purposes (Geijtenbeek et al., 2013). In other work, a simple yet fully adaptable form of evolvable musculature has been explored in which linear-spring-like muscles are evolved along with skeletal segments and control to produce true EVCs with some of the potential benefits of soft-body muscle actuation (Lessin et al., 2014b).

In this paper, a new approach is introduced in which an existing soft-body simulation system is re-purposed and exploited to produce realistic muscles with many desirable properties for the development of complex and capable EVC bodies (Figure 1a). In this way, evolved virtual creatures are produced which for the first time combine a traditional articulated skeleton with soft-body morphological elements in a new step on a nature-inspired path toward the type of morphological complexity that has proven so useful in the natural world.

Background

The primary contribution of this paper is a novel combination of traditional articulated-skeleton EVCs and soft-body simulation. This section provides a brief overview of these foundational technologies.

First, it is useful to define previous EVC systems. Classical EVCs were established in the initial work by Sims (1994b), then applied without major changes in a number of subsequent publications (Chaumont et al., 2007; Miconi, 2008; Lehman and Stanley, 2011). In these systems, the articulated rigid-body system is defined by a tree-like graph genotype, which is traversed to produce instructions for expressing the phenotype. The phenotype is typically composed of boxes or other rigid-body-simulation primitives (such as spheres and capsules), connected by a variety of joint types (such as revolute, spherical, prismatic, or cylindrical).

In the original implementation and most that followed, actuation was provided by implicit joint drives at every degree of freedom of every joint. In more recent work (Lessin et al., 2013, 2014b,a), these were replaced by linear-spring muscle activation, which is the implementation from which this paper's method is most directly derived.

While a number of implementations have employed neural networks for control (Miconi, 2008; Lehman and Stanley, 2011), Sims' original EVCs (Sims, 1994b,a) were controlled using a network of simple computing nodes (e.g., sinusoidal, sum, product, derivative), and that method is continued in the work of this paper. (See Figure 7b in the Results section for an example.)

It is also important to note previous work with soft bodies in EVCs. While they have never before been used in EVCs in combination with an articulated rigid skeleton, soft bodies have been applied to great effect in recent work to produce morphologically complex locomoting creatures, including some that use a combination of hard and soft body elements (though notably with the hard elements mixed in among the soft ones, not with any kind of articulated skeleton). Note that while those previous soft-body implementations employed single-purpose voxel-based soft body simulators (Cheney et al., 2013) or soft-body simulation alone (Rieffel, 2013), the work presented in this paper uses an adaptation of soft bodies within an integrated soft-and-rigid-body simulation system. This is necessary to permit the key combination of articulated-skeleton and soft-body simulation which is at the heart of this system's novel contribution.

Approach

To take this next step in nature-inspired morphological complexity for evolved virtual creatures, previous work on EVCs with linear-spring muscles will be combined with a novel soft-body muscle implementation. Beyond the new soft-body elements, most of the important details of the system (e.g.: joint and segment types, evolutionary algorithm, and genotype encoding and expression) are as defined in those earlier implementations.

In (Lessin et al., 2013, 2014b,a), a degree of morphological complexity was added through the use of a very simple approximation to a natural muscle: a linear spring. In that work, the traditional joint-motor drives employed by most EVCs were removed, and linear springs were allowed to evolve between skeletal segments. For each joint, springs could be added or removed by evolution, and their attachment points and strength were also evolvable. Brain activations applied to those muscles modified their underlying spring constant, increasing or decreasing the force they exerted on their attached body segments.

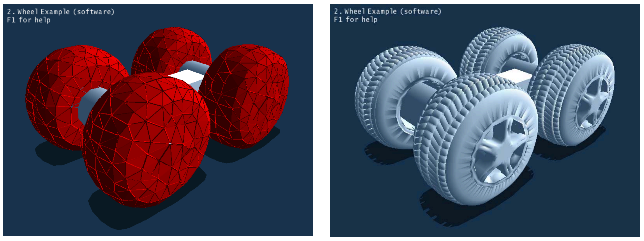
That work is the basis for the new system described here. Producing an appropriate soft-body muscle implementation is a significant challenge, but once that is achieved, it can be directly adapted to the linear-spring-muscle system just described. The system can function largely as before, simply constructing a full soft-body muscle in place of the original linear spring, using the given attachment points and strength. In the next section, the novel implementation of an appropriate soft-body-muscle system is described in detail.

Novel Soft-Body Muscle Implementation

The presented soft-body EVC musculature has a number of desired characteristics, each one obtained by employing or re-purposing existing capabilities available in an off-the-shelf physical simulation system. In this section, each of these elements of the system is described in detail.

Soft-Body Simulation

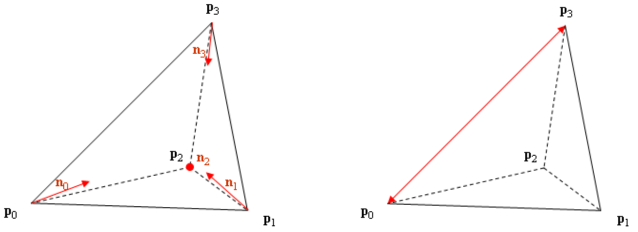
The fundamental soft-body system used is available as a part of NVIDIA PhysX, but not in most versions, and (at the time of writing) not in recent versions. (The work presented here was implemented with SDK version 2.8.1 and PhysX System Software 9.11.0621.) This system offers well-established abilities to simulate rigid bodies and joints, and can do so in combination with the soft-body simulations which are the focus of this work.



(a) (b)

Figure 2: PhysX's soft-body simulation is implemented using a tetrahedral mesh (a) which can be used to drive an accompanying triangular mesh for rendering (b).

In this system, soft bodies are simulated as tetrahedral meshes (Figure 2a), with vertices simulated as particles, and additional constraints applied per tetrahedron to produce some of the more complicated effects described in the following subsections. For rendering, the simulated tetrahedral mesh can be used to deform a corresponding triangle mesh of arbitrary complexity (Figure 2b).



(a) Tetrahedron vertices are moved to preserve volume, when *volume stiffness* is high.

(b) Tetrahedron vertices are moved to preserve edge lengths, when *stretching stiffness* is high.

Figure 3: Physx volume (a) and stretching (b) constraints, used to preserve muscle volume and implement muscle contraction, respectively.

Volume Preservation

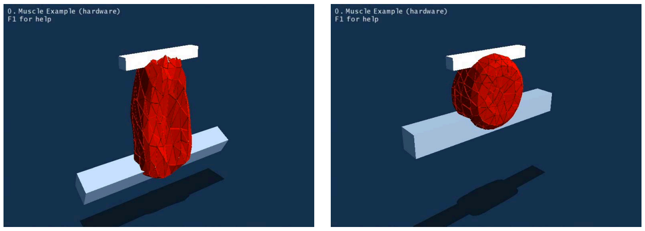
One desired characteristic of soft body muscles is the preservation of volume, which produces a familiar squash-and-stretch behavior as muscles decrease and increase in length. In PhysX, this property (which they call *volume stiffness*) is directly available as an attribute of simulated soft bodies, and is implemented as restorative movements applied to tetrahedral vertices (Figure 3a). This parameter may be varied to produce a range of soft-body styles. For this implementation, it was set to its maximum value: 1.0.

Contraction Mechanism

A vital aspect of simulated muscles is the ability to contract in a controllable manner. While PhysX provides no built-in mechanism for this, it does offer the *stretching stiffness* attribute of soft bodies, which can be repurposed to accomplish this goal. This property controls the degree to which a soft body will attempt to retain its original shape (again by changing tetrahedron-vertex positions), as illustrated in Figure 3b. By varying this attribute over time, the muscle's tendency to return to its initial, fully contracted shape is controlled, and it can be made to relax or contract as needed. In figure 4, the ability of this contraction mechanism to lift a rigid-body object is demonstrated.

Strength that Varies with Cross Section

Variations in muscle thickness add meaningful morphological complexity to animal musculature. To have the same kind of variation occur in a meaningful way in these virtual creatures, muscle strength can be made to depend on the muscle's cross-sectional area. A similar relationship was used in Sims' original work (Sims, 1994b), in which joint-motor strength varied in proportion to parent-segment cross section. With the soft-body simulations used for the muscles described here, this strength increases as a natural side-effect of the scaling process. By simply modifying the initial vertex positions of the tetrahedral mesh to give it thicker proportions, the muscle is able to pull with greater force when simulated by PhysX.



(a) Low stretching stiffness; muscle relaxed.

(b) High stretching stiffness; muscle contracted.

Figure 4: This figure illustrates the underlying contraction mechanism employed by the soft-body muscles in this paper. As stretching stiffness is increased, the simulated muscle changes from its relaxed shape (a) to its original, contracted shape (b), and is able to do useful work, such as lifting the rigid-body segment shown here.

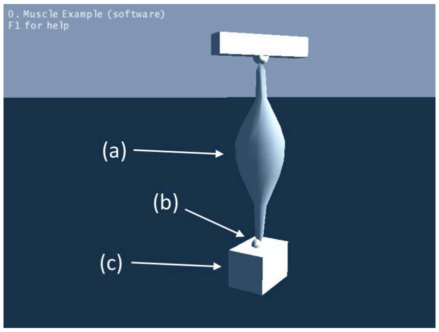
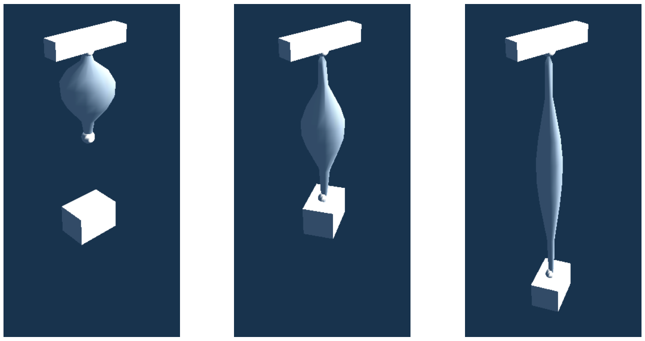


Figure 5: The tuberosity mechanism which allows muscles to realistically interact with adjacent skeletal segments: A muscle (a) is attached first to a tuberosity (b), which is in turn attached to the target skeletal segment (c).

Ability to Wrap Around Skeletal Segments

The fact that muscles wrap around skeletal segments in real-world bodies is also an important contributor to the complexity of animal musculature, but in the virtual world, such behavior is not a given. When soft bodies are used in PhysX, for example, any rigid body shape that they are attached to is prohibited from producing collision interactions with them. To get around this limitation, a small additional amount of bio-mimetic complexity is added during the muscle attachment process.

Instead of attaching simulated muscles directly to the rigid-body segments that they will act upon, each end of the muscle is first attached to a small sphere, which is then attached to the target segment (Figure 5). We refer to these spheres as *tuberosities* for their similarity to the eponymous structures in animal skeletons. With this configuration, although the virtual muscles are prohibited from colliding



(a) Fully contracted (before attachment).

(b) Pre-stretched for attachment.

(c) At maximum stretch when relaxed.

Figure 6: Muscle pre-stretch during construction. Muscles are instantiated in the fully contracted state (a), then pre-stretched to double their length for attachment to skeletal segments (b). During use, they may be contracted toward their original state, or further stretched (c) when relaxed.

with the tuberosity spheres themselves, collisions with all other skeleton segments occur as normal.

Muscle Pre-Stretch

The contraction mechanism described above requires the simulated muscles to be initially instantiated in their fully contracted state. To produce a neutral length during body construction, muscles are *pre-stretched* in the attachment process. Initially, muscles are created to span only half of the required distance between their attachment points (Figure 6a). Because of this, the attachment process serves to pre-stretch the muscle to an intermediate length (Figure 6b). From this attached pose, the muscle can be either contracted toward its initial shape, or further relaxed to its maximum length of approximately double the neutral length (a limit imposed upon soft bodies by PhysX), as shown in Figure 6c.

Experiments

Using the techniques described in the previous sections, a series of experiments were performed to evaluate this new system's potential to produce locomoting EVCs which combine for the first time an articulated skeleton with soft-body muscles.

Experimental Setup

For these experiments, eight independent runs were started simultaneously, each on its own processor in a relatively up-to-date laptop machine. Each run was started with a different random seed, and was allowed to run for approximately two weeks, using a population size of 50. In that time (apparently due to execution speeds which were highly dependent on creature complexity), the various runs completed between 180 and 893 generations. The simulation required for each fitness evaluation consisted of 1.5 seconds with the skeleton frozen (for muscles to recover from pre-stretch), up to 4 seconds for the body to settle (with the brain inactive), followed by the 6 seconds of simulation actually used to produce the fitness score.

Prevention of Cheating

Before presenting the results, it is important to describe an insidious form of cheating that was encountered, as well as what was done to prevent it. Many early tests fell victim to a particularly easy physics cheat apparently made possible by the muscles' soft-body simulation. It seems to be the case that with a sufficiently strong constant contraction of muscles, the sum of small simulation inaccuracies (made worse by the number of vertices simulated in these soft-body muscles) was sufficient to produce a physically unrealistic overall force on the creature. This could be used to generate movement along the ground with no physically realistic cause. Such solutions are apparently so prevalent in the solution space, that they completely excluded any non-cheating results when fitness was based solely on locomotion.

To overcome this cheating so that valid locomotion results could be discovered, an additional requirement was added: Each muscle should significantly change its length throughout the entire fitness evaluation, making the previous continuous, strong-contraction cheating technique impossible. Specifically, each muscle was required to change its length by 25% during each 0.5 seconds of evaluation. During speed evaluation, this is strictly enforced, with zero fitness assigned to any individual not sufficiently achieving this goal. But it is also not trivial to produce a creature in which this requirement is accomplished. This challenge is met by introducing a two-stage evolutionary process, with the first stage providing a fitness gradient toward the muscle-length-change goal, and the second stage rewarding locomotion among creatures which continue to meet that initial requirement.

For stage one, fitness is calculated as the average across all muscles and all 0.5-second evaluation periods of how well the 25% length variation is achieved. So, if E is the set of evaluation periods, M is the set of the creature's muscles, and l_{min} and l_{max} are a muscle's minimum and maximum length during an evaluation period, stage-one fitness f_1 can be computed as

$$f_1 = \frac{1}{|E|\,|M|} \sum_{E,M} max(1, \frac{l_{max} - l_{min}}{l_{max}} \cdot \frac{1}{0.25}).$$

(Note: The $max()$ term ensures that the required fitness is developed for all muscles in all evaluation periods. Without it, the same overall f_1 fitness might be achieved with some muscles having more than the required minimum and others having less, or none.)

Once a sufficient fraction of the population (5% was used in these experiments) has achieved full fitness using f_1, the population is refilled completely with copies of those f_1-fit individuals, and stage two begins.

In stage two, in addition to f_1, speed is also evaluated, and used with f_1 as described earlier. If full f_1 fitness is maintained (or nearly so), the new fitness f_2 is simply computed from speed. But if f_1 fitness falls too low, f_2 will be assigned a zero value. If s is the average speed during the evaluation, and s_{max} is the maximum target speed (20 m/s for these experiments), f_2 can be computed as

$$f_2 = \begin{cases} \frac{s}{s_{max}} & \text{if} f_1 > 0.9 \\ 0 & \text{otherwise} \end{cases} .$$

Results

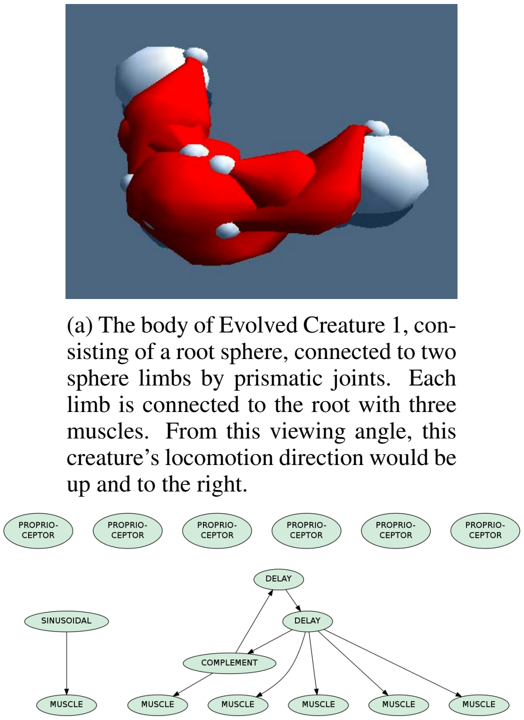
Of the eight experimental runs, four were successful with respect to fitness score, producing stage-two scores associated with reasonable-looking locomotion. One of those four succeeded by exploiting a weakness in the fitness definition rather than by producing true locomotion. The remaining three successful runs (runs 3, 4, and 6) are presented here as Evolved Creatures 1 through 3. To see these results in motion, please visit `http://goo.gl/rvSvFv` .

Evolved Creature 1: Crawler

In this section, the first example of a successful result (run 3, generation 172) is presented. Its body and brain are shown in Figure 7. This creature's rigid skeleton consists of a central sphere attached to spherical limbs by prismatic joints (able to telescope linearly, while maintaining orientation). Each limb is connected to the root segment by three muscles, with varying attachment points and strengths.

This creature's brain shows that it employs open-loop control—it ignores input from its proprioceptors (sensors indicating muscle length) when computing the output signals sent to the muscles. Also notable in this brain: One muscle is driven directly by a sinusoidal signal, while the rest of the brain nodes have been employed to create an improvised square-wave generator which drives the remaining muscles.

As is true for all of the results presented here, this creature's morphology makes use of bilateral symmetry, accessible to evolution as a single boolean attribute. Its method of locomotion is to move forward (toward its limbs) by quickly extending and retracting its arms in an alternating manner, producing a clawing, crawling gait (Figure 9).



(a) The body of Evolved Creature 1, consisting of a root sphere, connected to two sphere limbs by prismatic joints. Each limb is connected to the root with three muscles. From this viewing angle, this creature's locomotion direction would be up and to the right.

(b) The brain of Evolved Creature 1.

Figure 7: The body and brain of the first example of a successful result from this system. This creature was produced at generation 172 of evolutionary run 3.

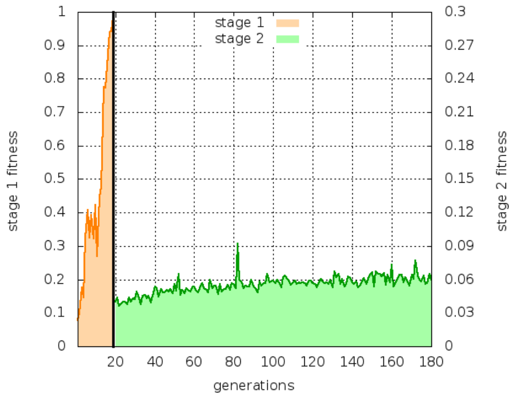


Figure 8: Fitness graphed over time for the evolutionary run producing Evolved Creature 1. The graph shows the development of stage-one fitness on the left in orange (in which muscle-length changes are rewarded) and stage-two fitness on the right in green (in which locomotion is rewarded, and muscle-length changes are required).

The development of this creature's fitness is shown in Figure 8. Generations in stage one (learning to keep muscle lengths changing) are shown on the left of the graph in red, with the generations spent in stage two (learning locomotion while maintaining muscle movement) shown on the right of the graph in green. In this creature, the ability to change all muscle lengths and complete the first stage was achieved very quickly, followed by slow, steady optimization of control with no obvious morphological changes.

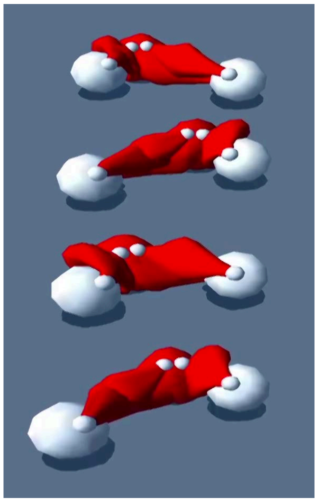
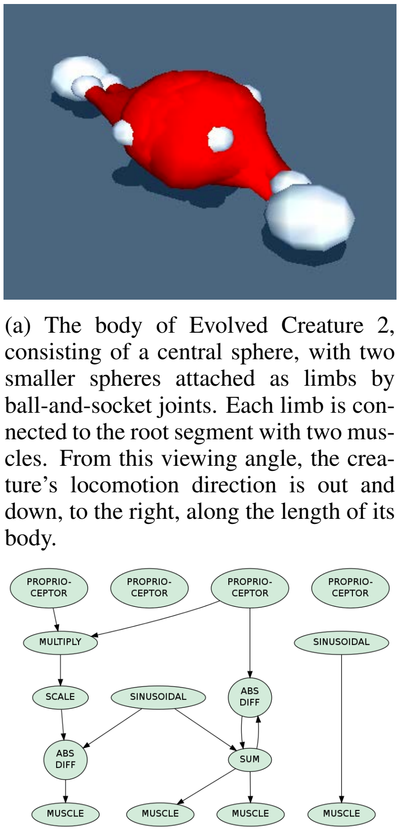


Figure 9: The locomotion method of Evolved Creature 1. Viewing the images from top to bottom, the creature is seen to shift its weight from side to side, extending alternating limbs forward (toward the viewer) to produce a crawling gait in that direction.

Evolved Creature 2: Inchworm

The next successful result presented (run 4, generation 518) is discussed in this section. Its body and brain are illustrated in Figure 10. Its skeleton consists of a large central sphere, joined to two smaller spheres by spherical (ball-and-socket) joints. The central segment is connected to each of its limbs by two muscles. In contrast to the previous creature, this creature's brain is harder to interpret. It is notably different, however, in that it appears to employ some closed-loop control, having connections from its proprioceptive sensors into the rest of the control network.

By aggressively swinging its central segment toward one of the limbs, then more gently returning it towards the other, this creature locomotes along the direction of its body length (Figure 12), producing the fastest creature by far among the results presented in this paper. As before, both stages of this creature's fitness development are illustrated in Figure 11. In contrast to the first creature, this creature spent many more generations developing full stage-one fitness (rewarding muscle-length change), and its stage-two fitness makes significant jumps after appearing to level off on two separate occasions.



(a) The body of Evolved Creature 2, consisting of a central sphere, with two smaller spheres attached as limbs by ball-and-socket joints. Each limb is connected to the root segment with two muscles. From this viewing angle, the creature's locomotion direction is out and down, to the right, along the length of its body.

(b) The brain of Evolved Creature 2.

Figure 10: The body and brain of the second example of a successful result from this system. This creature was produced at generation 518 of evolutionary run 4.

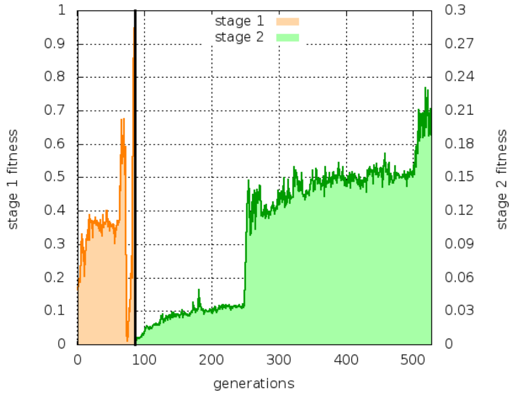


Figure 11: Fitness graphed over time for the evolutionary run producing Evolved Creature 2.

Evolved Creature 3: Shuffler

In this section, the final successful result (run 6, generation 348) is presented, with body and brain illustrated in Figure 13. This creature's skeleton is composed of a central capsule segment, with capsule limbs extending out and to the back. The limbs are connected with cylindrical joints, which allow both telescoping along the joint's axis and rolling around it, as well. Each limb is connected to the central segment by two muscles. By shifting its weight from side to side, forward (away from the limbs) shuffling locomotion is produced. The two stages of fitness development are shown in Figure 14. In contrast to the two previous crea-

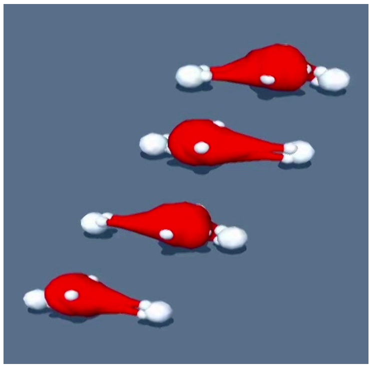
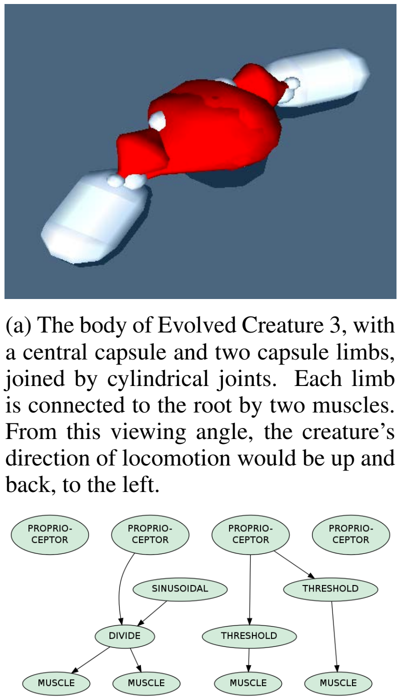


Figure 12: The locomotive technique of Evolved Creature 2. Considering the images from top to bottom, the creature first draws its weight back (to the right in this view), then aggressively throws it forward (left in this view) producing an inchworm-like gait along the direction of the length of its body.



(a) The body of Evolved Creature 3, with a central capsule and two capsule limbs, joined by cylindrical joints. Each limb is connected to the root by two muscles. From this viewing angle, the creature's direction of locomotion would be up and back, to the left.

(b) The brain of Evolved Creature 3.

Figure 13: The body and brain of the third example of a successful result from this system. This creature was produced at generation 348 of evolutionary run 6.

tures shown, this creature spent over half of its simulation time developing stage-one fitness.

Results Summary

The experiments presented here have demonstrated that this paper's novel soft-body muscle system is indeed sufficient to allow, for the first time, the evolution of virtual creatures which combine a rigid-body skeleton with complex non-rigid muscles. These evolved results demonstrated multiple locomotion solutions, employing a variety of segment types, joint types, control strategies, and gaits.

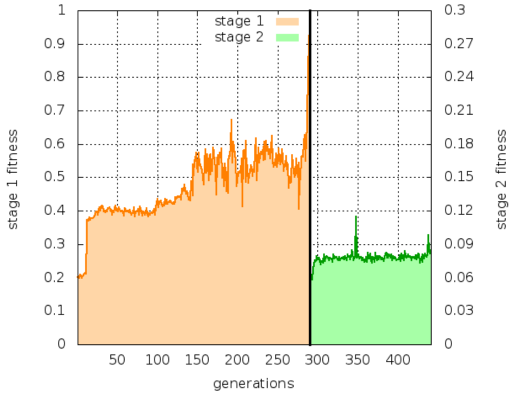


Figure 14: Fitness graphed over time for the evolutionary run producing Evolved Creature 3.

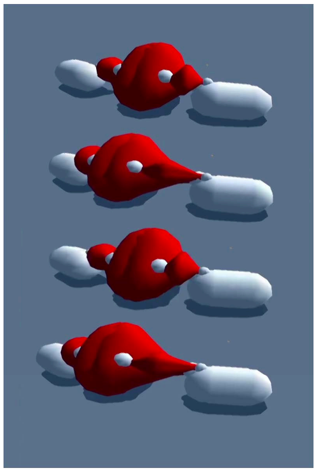


Figure 15: The locomotive technique of Evolved Creature 3. From top to bottom, the images show how the creature shifts its weight from side to side, resulting in a forward-sliding, shuffling gait. From this viewing angle, the locomotion direction is out, down, and slightly to the left.

Discussion and Future Work

One important issue to discuss is the fact that the anti-cheating mechanism employed (requiring muscle-length changes) means that some otherwise valid solutions may never be evolved. This was an acceptable trade-off for these initial experiments, but in the future, a more accurate physical simulator or different method of preventing cheating might allow this restriction to be removed.

One observation about the results presented here is that, despite the demonstrated variety in body structure, brain structure, and gaits, all of these successful creatures share a similar body plan, with a limited number of segments in somewhat similar configurations. Future work might determine the cause for this or get beyond it, perhaps through the use of a diversity-promoting mechanism (Lehman and Stanley, 2011).

In the larger picture, the work presented here is part of a potentially rewarding path to the kind of rich morphological complexity seen in creatures in the natural world. Now that bodies can be evolved with functional muscles which contribute in a meaningful way to morphology, one clear next step would be the addition of simulated skin. (In fact, this was even proposed in Sims' original work). The same kinds of physical simulation systems that provide soft-body simulation can also simulate cloth, which might be an ideal way to achieve this next layer of bio-mimetic realism in virtual creatures.

Another compelling topic on the same path might be the evolution of bone shapes, as enabled by the ability to effectively evolve three-dimensional forms (Clune and Lipson, 2011). Not only would this allow richer morphological interactions with muscles and skin, but it might even permit the development of emergent joints. Instead of using implicit, externally enforced relationships to define the relative movement of connected segments, evolvable bone shapes might allow these relationships to emerge naturally from the interactions between bones, muscles, and skin. Additionally, evolvable bone shapes might permit the development of creatures with exoskeletons, which may well emerge naturally through this process as long as they are not explicitly disallowed.

Conclusion

This paper has presented a technique for combining rigid articulated skeletons with soft-body muscles in evolved virtual creatures for the first time. This was made possible by a novel combination and re-purposing of a number of existing physical simulation components from an off-the-shelf simulation system, as detailed above. Initial experiments with evolving morphology and control for locomotion were shown, along with a new method for counteracting evolution's destructive exploitation of this simulator's inaccuracies. These experiments had a relatively high rate of success in producing locomoting creatures, and a number of useful results were presented. This novel bio-mimetic synthesis of two highly successful EVC techniques represents a new step toward matching the morphological complexity of some of the most successful creatures in the natural world.

References

Auerbach, J. E. and Bongard, J. C. (2012). On the relationship between environmental and morphological complexity in evolved robots. In *Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference*, GECCO '12, pages 521–528, New York, NY, USA. ACM.

Chaumont, N., Egli, R., and Adami, C. (2007). Evolving virtual creatures and catapults. *Artificial Life*, 13(2):139–157.

Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: Evolving soft robots with multiple materials and a powerful generative encoding. In *Proceeding of the Fifteenth Annual Conference on Genetic and Evolutionary Computation Conference*, GECCO '13, pages 167–174, New York, NY, USA. ACM.

Clune, J. and Lipson, H. (2011). Evolving 3d objects with a generative encoding inspired by developmental biology. *ACM SIGEVOlution*, 5(4):2–12.

Geijtenbeek, T., van de Panne, M., and van der Stappen, A. F. (2013). Flexible muscle-based locomotion for bipedal creatures. *ACM Trans. Graph.*, 32(6):206:1–206:11.

Glette, K. and Hovin, M. (2010). Evolution of artificial muscle-based robotic locomotion in PhysX. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 1114–1119. IEEE.

Lehman, J. and Stanley, K. (2011). Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM.

Lessin, D., Fussell, D., and Miikkulainen, R. (2013). Open-ended behavioral complexity for evolved virtual creatures. In *Proceeding of the Fifteenth Annual Conference on Genetic and Evolutionary Computation Conference*, GECCO '13, pages 335–342, New York, NY, USA. ACM.

Lessin, D., Fussell, D., and Miikkulainen, R. (2014a). Adapting morphology to multiple tasks in evolved virtual creatures. In *Proceedings of The Fourteenth International Conference on the Synthesis and Simulation of Living Systems (ALIFE 14) 2014*.

Lessin, D., Fussell, D., and Miikkulainen, R. (2014b). Trading control intelligence for physical intelligence: Muscle drives in evolved virtual creatures. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2014*.

Miconi, T. (2008). In silicon no one can hear you scream: Evolving fighting creatures. *Genetic Programming*, pages 25–36.

Moore, J. M. and McKinley, P. K. (2014). Evolving joint-level control with digital muscles. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 209–216, New York, NY, USA. ACM.

Rieffel, J. (2013). Heterochronic scaling of developmental durations in evolved soft robots. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation*, GECCO '13, pages 743–750, New York, NY, USA. ACM.

Sims, K. (1994a). Evolving 3d morphology and behavior by competition. *Artificial life*, 1(4):353–372.

Sims, K. (1994b). Evolving virtual creatures. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, SIGGRAPH '94, pages 15–22, New York, NY, USA. ACM.

Cascading Transitions in Coupled Complex Ecosystems

Iain S. Weaver

University of Southampton, University Road, Southampton, SO17 1BJ
iainweaver@gmail.com

Abstract

The Earth system is arguably one of the most complex systems in the known universe. Over 4.5 billion years it has self-organised into a state which features complex differentiated life that covers its surface and penetrates its crust. This widespread biosphere requires stability in the sense of maintenance of temperature and pressures on the surface that allow liquid water. Within this range there is significant scope for change, some of it dramatic. Glaciation to inter-glacial cycles are classic examples of planetary-scale transitions. This paper examines the role of biological feedback on planetary-scale transitions. We present a conceptual model in which stability emerges as a consequence of interactions between environment and life. These mechanisms not only lead to stable ecosystem configurations, but can also produce critical transitions which we characterise as cascading transitions. Such failures can interact and produce system-wide transitions. Our results would be of interest to those studying real-world ecosystems, tipping elements in the Earth system as well as theoretical studies on complex artificial life systems.

Introduction

Tipping points, where critical levels of stress poise a system on the precipice of a transition between its current state and another, are a common feature of many complex systems with at least bi-stability (Lenton et al., 2008). Examples are numerous across all scales and areas of science, from the earliest studies of phase transitions in physics to multi-agent social, ecological and other life-environment systems (Stanley, 1987; Holme and Newman, 2006; Chavez et al., 2013; Williams and Lenton, 2007). This paper is centred on relating local Earth system transition on the ecosystem scale, to planetary scale transitions.

It has even been proposed that, on the planetary-scale, the state of planet Earth may approach a tipping point where a critical point in a component of the Earth-system corresponds to a global tipping point such as by a domino effect, or ‘tipping cascade’, as the result of forcing effects from human activity (Barnosky et al., 2012). In such a case, the global ecology undergoes a catastrophic shift to some new, unpredictable composition. Indeed, Earth’s history is punctuated by such transitions; the Snowball Earth hypothesis posits a massive glaciation event 650 million years ago, where it is thought that critical levels of sea-ice overcame the previously stable global conditions, causing a transition to an almost completely frozen state (Hoffman and Schrag, 2002). The hypothesis is aptly named as a description both of the state of the planet, and the runaway positive feedback mechanism thought to be responsible for such a catastrophic transition.

The principle of tipping points is less contentious when applied to individual Earth-system components and local ecosystems although it remains an ever more pressing issue (Lenton et al., 2008); pressure on the Earth system from humans, such as anthropogenic emissions, climbs ever higher, and the response of the components of the Earth system appear unpredictable. To what extent do we expect complex systems to maintain robustness in the face of increasing perturbations, how can we characterise the behaviour of near-critical systems, and how do Earth-system components reorganise when pushed beyond their regulatory limits? In part, the difficulty in answering these key questions stems from the absence of any broad real-world mechanisms which govern the stationary and critical behaviour of Earth-systems.

It is clear that there is some degree of coupling between the biotic and abiotic components of Earth. This is particularly evident in large systems, such as Earth’s atmosphere. While roughly stable, they exist far from chemical equilibrium; the levels attained from purely abiotic processes (Kleidon, 2011). The living elements of these systems carry out a crucial role in establishing this stationary behaviour. However, non-equilibrium systems require balance to establish stability, and disruptions to this either through external perturbation, or large internal fluctuations, may result in catastrophes; periods of rapid change for the composition of the biota, and the state of its environment (Lenton et al., 2008). This can happen when environmental conditions change abruptly, such that the biota exerts an overall effect in its environment, and this has a positive influence on the activity of the agents of change, the previously stationary behaviour is lost, and the configuration both of the biota and its environment approach a new attractor. In the case of lake eutrophication, the system can be shifted into an

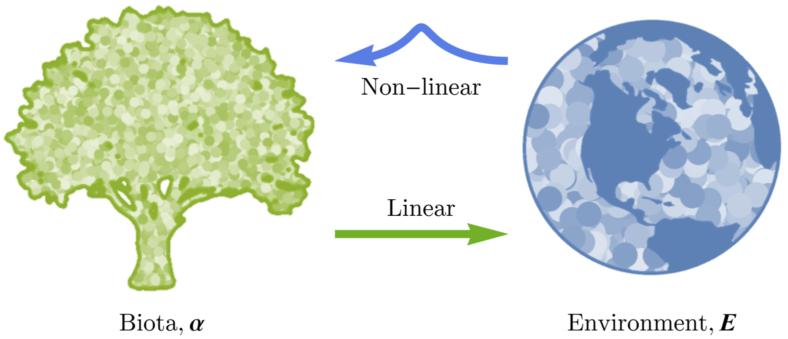


Figure 1: The model biota consists of a large number, K, elements in the vector $\boldsymbol{\alpha}$. They are linearly coupled to their N environmental variables in vector $\boldsymbol{E}$ by the random, zero-mean coupling coefficients $\boldsymbol{\Omega}$. In turn, the environment determines their individual abundance by a simple niche idea; biotic elements are maximally abundant in the vicinity of their niche, $\boldsymbol{\mu}$.

anoxic state by increasing concentrations of otherwise limiting nutrients; while many plants and algae may flourish, the deoxygenation which results from their decomposition strongly favours the disruptive algae whose proliferation further alters the conditions to their relative favour.

Lenton and Williams (2013) discuss conditions under which this mechanism may lead to to global transitions, dubbed 'planetary-scale tipping points'. These phenomena can occur when very large, and generally well-mixed systems undergo such a transition. Changes in the composition of the atmosphere, for example, is expected to have a rapid and global effect on the Earth system. Another mechanism by which the global state may abruptly shift is in response to some uniform, global pressure which causes a wide range of Earth-systems to transition simultaneously. Lenton and Williams (2013) point out the improbability of this explanation which would require a universal ecological threshold in spite of the inhomogeneity in the global ecology. The more likely alternative is that transitions in Earth systems on the local scale influence connected, codependent systems. By this process, the global system may self-organise such that local transitions can cause cascades, where failures of the regulatory mechanism in connected systems coincide not by chance but due to a causal or 'domino' effect. A sudden change in the environmental state of one system undergoing a transition will impact connected systems. If this is sufficient to destabilise the connected system, this can lead to an avalanche of transitioning systems, resulting in continental- and planetary-scale transitions.

Can transitions propagate through connected ecosystems in this way? The aim of this paper is to study a meta-system produced by the coupling of a large number of simple model ecosystems, whose coupling enables them to effect local transitions in one another. These models may be seen as a generalisation of Watson and Lovelock's (1983) Daisyworld; the two daisy types are replaced by a large and diverse biota while the simple, one-dimensional environment has been replaced by one of arbitrary dimensionality (Weaver and Dyke, 2013). We begin by describing the model in four components;

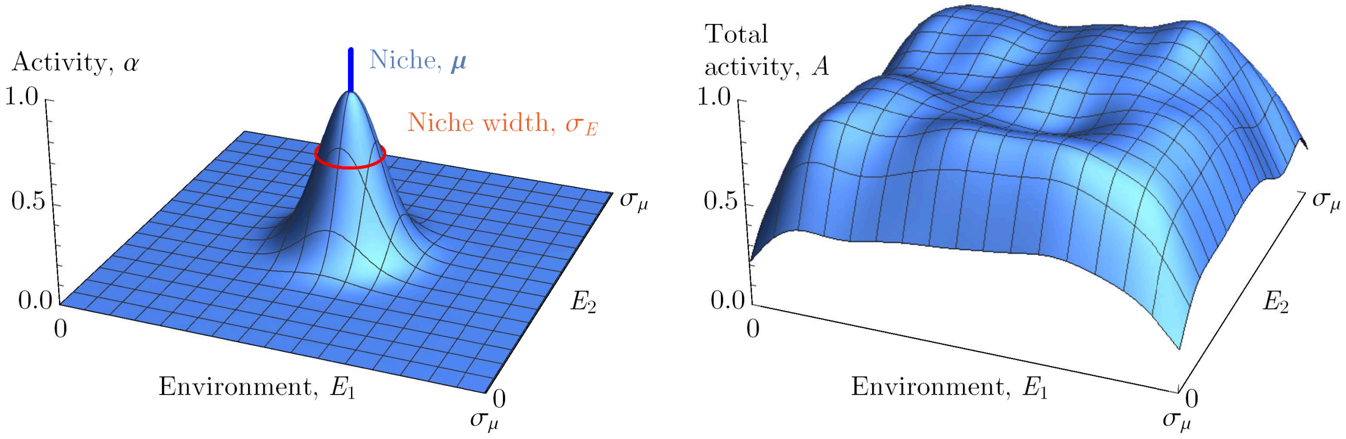
i) abiotic model elements and their influence on the biota,
ii) biotic elements and their effect on the environment,
iii) external perturbing factors which affect the environment,
iv) local coupling between adjacent systems defines the meta-system.

Following this, we examine the effect of local transitions on a network of complex ecosystem models with a range of connection strengths. The paper concludes with a discussion of the main results, along with a comparison to other complex systems which display a diversity of fixed points and internal transitioning behaviour.

Model formulation

The model can be viewed as a simplification of Watson and Lovelock's (1983) Daisyworld model which foregoes the need to precisely prescribe the behaviour of the small biota, instead favouring a large ensemble of randomly parametrised elements, similarly to Dyke (2010). The model components are summarised by Fig. 1, and explained in detail in this section. The biota is represented by a number K of independent and randomly parametrised biotic elements. We avoid making any assumption about the form the biota takes; be they individual organisms, species, ecotypes or populations. Furthermore, the biota is considered well-mixed and possesses no organisation or structure in and of itself; a cogent feature of some other artificial ecosystems (Chavez et al., 2013). Their individual abundance, metabolic activity or overall influence is denoted in the elements of the vector $\boldsymbol{\alpha}$. Importantly, the biota lacks any form of self-interaction through processes such as competition or predation.

These elements interact collectively with a simple environment, comprised of N variables denoted by the vector $\boldsymbol{E}$. As before we do not prescribe a physical interpretation of the environmental dimensions beyond asserting they are variables which are both influenced by the biota, and influence the composition of the biota. The effect of individual biotic elements is unidirectional and monotonic;



(a) The activity of a biotic element with a niche in the center of the essestial range.

(b) The total activity of many biotic elements witha diversity of niches.

Figure 2: The activity of an element of the biota is a function of its environment $\boldsymbol{E}$ and niche position $\boldsymbol{\mu}$ are shown in Fig. 2a for an $N = 2$-dimensional environment. The precise choice of function has been shown to be arbitrary with only the characteristic width, σ_E being important (Weaver and Dyke, 2013). As the essential range is populated, the total activity shown in Fig. 2b approaches uniformity.

each species has only an increasing or decreasing effect on each aspect of the environment which is directly proportional to their biotic activity. These minimal ingredients, encapsulated in Fig. 1, have been shown to produce a range of interesting behaviours (Dyke and Weaver, 2013). Most significantly, homeostatic fixed points in the environmental variables emerge over a wide range of assumptions merely from the effects of a randomly assembled biota, foregoing the need for parametrisation or tuning. Feedback mechanisms exist within the environment in real systems; chemical weathering of silicate rocks is a negative environmental feedback where increases in global temperature, for example by increases in atmospheric CO_2 levels, lead to increased silicate rock weathering, a process which captures CO_2. However, this model lacks such environment-environment or life-life feedbacks; any coupling between biotic elements is mediated through the environment and vice-versa. Such interactions are excluded to avoid prescribing which limiting factors may be relevant. Competition is an example of feedback within the biota but is only one of many potential factors; on the biospheric level, the majority of biotic elements may not directly interact at all and so relaxing the assumption of strong interactions between biotic elements broadens our results and analysis.

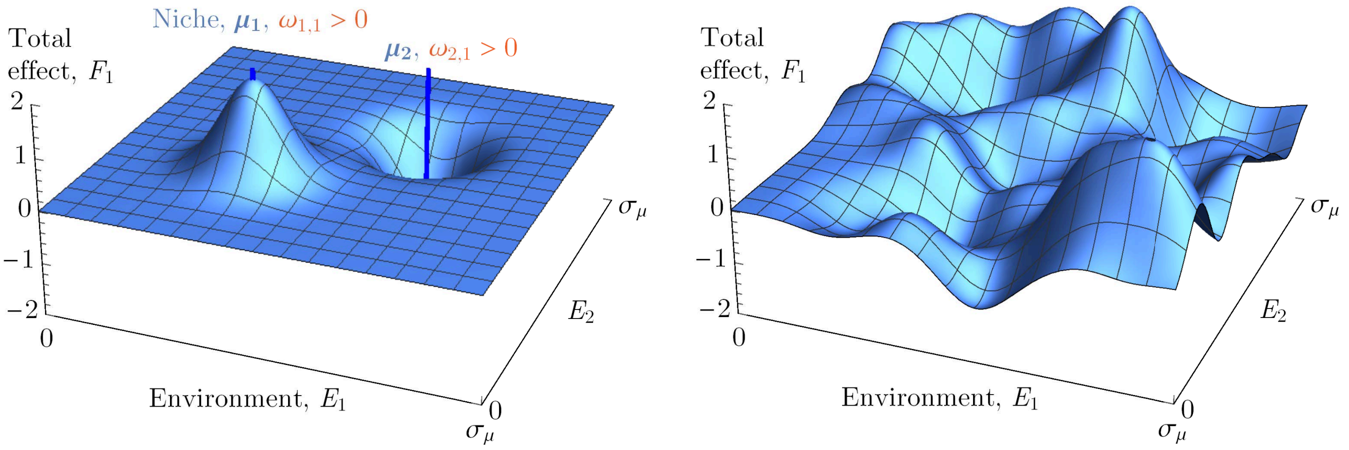
i) Environment affects life The first component of Fig. 1 we discuss is the influence of the state of the environment $\boldsymbol{E}$ on the composition of the biota $\boldsymbol{\alpha}$. We implement a very simple niche idea whereby each element of the biota has an optimal environment, a point in the space of environmental variables where its activity is maximised. The time evolution of each element of the biota is, in the simplest case, a linear relaxation towards some steady-state activity $\boldsymbol{\alpha}^*$ with characteristic time scale τ_α.

$$\tau_\alpha \frac{\mathrm{d}\boldsymbol{\alpha}(t)}{\mathrm{d}t} = \boldsymbol{\alpha}^*(\boldsymbol{E}, \boldsymbol{\mu}) - \boldsymbol{\alpha}(t) \tag{1}$$

where $\boldsymbol{\alpha}^*$ is the fixed point in the time evolution of $\boldsymbol{\alpha}$, dictated by the state of the environment and a niche parameter unique to the each biotic element $\boldsymbol{\mu}$, which determines the most advantageous environmental conditions for each biotic element. In the vicinity of $\boldsymbol{\mu}$, $\boldsymbol{\alpha}^*$ is maximised and decays by some function as the environment departs the niche. Niches are randomly distributed uniformly in a finite volume of environmental variables called the essential range, which imposes limits on the ability of the biota to prosper in extreme environmental conditions. Again, in the simplest case this range is equal in all environmental variables and denoted σ_μ. For the purpose of this paper, the precise choice of $\boldsymbol{\alpha}^*$ can be shown to be arbitrary (Weaver and Dyke, 2012; Dyke and Weaver, 2013) and we choose a N-dimensional Gaussian for visualisation purposes where appropriate although a parabola, step-function or some multi-modal function would equally suffice providing it has a well defined variance.

$$\alpha_i^*(\boldsymbol{E}, \boldsymbol{\mu}) = \exp\left(-\frac{(\boldsymbol{\mu}_i - \boldsymbol{E})^\top(\boldsymbol{\mu}_i - \boldsymbol{E})}{2\sigma_E^2}\right) \tag{2}$$

where σ_E is the niche width, the characteristic width of the function, illustrated in Fig. 2a. As the essential range be-



(a) The effect function generated by two biotic elements with opposing coupling values.

(b) The total effect function of many randomly parametrised biotic elements.

Figure 3: Two biotic elements with an increasing ($\omega_1 > 0$) and decreasing ($\omega_2 < 0$) effect on one environment variable are illustrated by Fig. 3a along with their niche positions. As the number of such elements, K, becomes large, the essential range is populated by a diverse biota and the net effect maintains zero mean, but significant variance, illustrated by Fig. 3b.

comes populated by a diverse biota, the mean activity increases with K, while the deviations increase with $\sqrt{K}$ such that in the limit of a very diverse biota of many unique elements, the total steady-state activity is approximately uniform over the essential range, shown by Fig. 2b.

ii) Life affects environment To be alive necessitates having some influence on the local environment through consumption or excretion of resources to maintain a metabolism although organisms may influence other environmental factors such as temperature, surface albedo and the abundance of other chemicals through their influence on other abiotic processes like weathering and erosion. Examples of such processes on Earth are abundant over nearly all spatial scales; from local effects such as changes in soil composition to modifications of atmospheric composition globally, by oxygenic photosynthesis (Wilkinson et al., 2009; Sanders and van Veen, 2011; Goldblatt et al., 2006). These impacts are encapsulated by concepts of niche construction and ecosystems engineering (Jones et al., 1994; Odling-Smee et al., 2003).

In the simplest case, the effects are linear and simply proportional to the activity of the biota, implemented by assigning each biotic element a unique influence on each aspect of its environment independently, stored in the matrix $\boldsymbol{\Omega}$ such that the summed effect on each environmental variable, $\boldsymbol{F}$ may be found simply from the matrix product

$$\boldsymbol{F}(t) = \boldsymbol{\Omega} \cdot \boldsymbol{\alpha}(t). \tag{3}$$

The effects in $\boldsymbol{\Omega}$ are chosen randomly with zero mean such that the model has no propensity for positive or negative feedback. The steady-state effects of two opposing biotic elements are shown by Fig. 3a, while the net effect of a large population is shown by Fig. 3b. Unlike the steady-state activity of a large population, the net effect of a population has zero mean and with appropriate scaling (discussed shortly) the variance in the surface is significant. The time evolution of the environmental variables is, in the simplest case, linearly driven by contributions from the biota $\boldsymbol{F}$ and external influence from adjacent systems, $\boldsymbol{\mathcal{N}}$, and global, abiotic influences $\boldsymbol{P}$, whose form is discussed shortly.

$$\tau_E \frac{\mathrm{d}\boldsymbol{E}(t)}{\mathrm{d}t} = \boldsymbol{F}(t) + \boldsymbol{\mathcal{N}}(\boldsymbol{E}) \tag{4}$$

The linear time evolution of $\boldsymbol{\alpha}$ ensures this system has a $2N$-dimensional phase space (Weaver and Dyke, 2013) though this can be reduced to N-dimensions if the time scales of processes associated with changes in the biota, $\boldsymbol{\alpha}$ and the environment, $\boldsymbol{E}$ may be assumed separated, such that changes to the ecology occur on very much shorter time scales than those to the environment, or $\tau_\alpha \ll \tau_E$. In this instance, the activity of the biota may be replaced by its steady-state value $\boldsymbol{\alpha}^*$.

It is useful to ensure that important model characteristics are invariant with the arbitrary model parameters, those which are neither random nor fundamental model components. This enables direct comparison of model behaviour over a range of parameters which might otherwise be obfuscated. We rescale the effect of the biota by introducing the normalisation constant

$$(\boldsymbol{\Omega} \cdot \boldsymbol{\alpha}) \rightarrow C^{-1}(\boldsymbol{\Omega} \cdot \boldsymbol{\alpha}) \tag{5}$$

where we have used the constant C to normalize the total biotic effect, $\boldsymbol{F}(\boldsymbol{E})$ such that its variance is independent of our choice of variance of the random coupling parameter, σ_Ω, the fundamental niche width, σ_E and the number of biotic components, K.

$$C^2 = 2K\sigma_\Omega^2 \left(\sqrt{\pi}\sigma_E\right)^N . \tag{6}$$

iii) Ecosystem connections Our model consists of a number of these systems connected into a network of interacting sub-systems, each with their own set of homeostatic states. In the absence of any coupling, where $\mathcal{N} = \mathbf{0}$, such a model behaves as a large number of independent systems whose large-scale statistics have already been established in depth (Dyke and Weaver, 2013). In this model, we couple adjacent systems by allowing the environmental variables to diffuse across edges with the following equation

$$\mathcal{N}(\boldsymbol{E}) = D_E \sum_{j \in \mathrm{nn.}} \left(\boldsymbol{E}_j - \boldsymbol{E}\right) \tag{7}$$

where D_E parametrises the coupling strength and $\mathrm{nn.}$ denotes the set of n neighbouring vertices. This coupling can be seen as a site being perturbed by its neighbours in direct proportion to the difference between the state of its environment and that of the neighbouring site and the factor $\frac{1}{n}$ ensures this forcing is not more or less influential on vertices with higher or lower than average degree. This term is similar to the 'leakage' or heat conductance described by Harvey (2004), a modification to the original Daisyworld model where populations of biotic elements have distinct but coupled environments. Indeed, the imposition of a network structure coupling distinct artificial ecosystems is not new (Punithan and McKay, 2013) although a key difference in this work is our focus not on large-scale pattern formation, but the propagation of local transitions through a connected meta-system.

To establish invariance between the behaviour of our coupled ecosystem models and our parametrisation, it is useful to define the coupling strength D_E in terms of the width of the effect function, σ_E. We have previously ensured the variance of the effect function to be unity, and so we define D_E in terms of the expected difference between homeostatic fixed points at a certain perturbation level

$$\mathcal{E} = 2\sqrt{2}\,\sigma\pi. \tag{8}$$

By this definition, a system undergoing a transition with $D_E = \frac{1}{\mathcal{E}}$ increases the magnitude of its diffusion to neighbouring sites by approximately unity.

It is important to note that we have not explicitly included the diffusion of our biota. The reason for this is that if we choose such a diffusion process to be linear with the biotic activity then the inclusion of another diffusion process can be shown to do nothing beyond modify the existing diffusion process. To show this, we start from Eq. (1), taking the product with the coupling matrix, $\boldsymbol{\Omega}$ as shown in Eq. (3), modifying the equation to include a diffusion term exactly as Eq. (7). In the steady state, we have

$$\boldsymbol{F}^*(\boldsymbol{E}) - \boldsymbol{F}(t) + D_F \sum_{j \in \mathrm{nn.}} \big(\boldsymbol{F}_j(t) - \boldsymbol{F}(t)\big) = 0 \tag{9}$$

where $\boldsymbol{F}^*(\boldsymbol{E})$ is the steady state value of $\boldsymbol{F}(t)$ is the absence of diffusion, where $D_F = 0$. Solving Eq. (9) for $\boldsymbol{F}(t)$ gives

$$\begin{aligned}\boldsymbol{F}(t) &= \frac{\boldsymbol{F}^*(\boldsymbol{E}) + D_F\big(\boldsymbol{F}^*(\boldsymbol{E}) + \sum_j \boldsymbol{F}^*_j(\boldsymbol{E}_j)\big)}{1 + D_F + nD_F} \\ &= \frac{\boldsymbol{F}^*(\boldsymbol{E})}{1 + D_F + nD_F}\end{aligned} \tag{10}$$

where n is the number of neighbours to the system, and we have eliminated the term including the summation which evaluates to zero at the steady state. Substituting this into the steady state condition for Eq. (4) gives

$$\boldsymbol{F}^*(\boldsymbol{E}) = (1 + D_F + nD_F)D_E \sum_{j \in \mathrm{nn.}} \big(\boldsymbol{E}_j - \boldsymbol{E}\big) \tag{11}$$

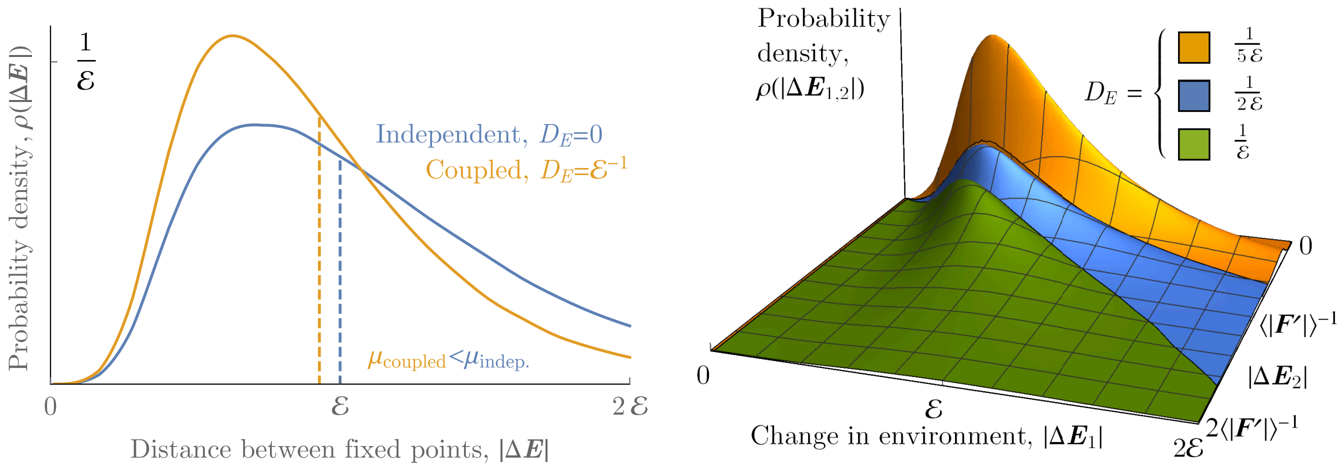
which is identical to Eq. (4) with the substitution of $\boldsymbol{F}(\boldsymbol{E})$ for its steady state value and D_E for the effective diffusion rate $(1 + D_F + nD_F)D_E$. In summary, if a linear relaxation is chosen for the change in the composition of the biota towards its steady statue value, and the time scale of this process is much shorter than changes to the environmental state, any diffusion term in this process may be absorbed into the environmental diffusion process giving an effective diffusion rate, D'_E, of

$$D'_E = (1 + D_F + nD_F)D_E \tag{12}$$

where D_F and D_E are diffusion rates of the biota and environment respectively.

Analysis

The life-environment systems detailed in the previous section have proven themselves to possess interesting behaviour in their own right; they generate a number of homeostatic fixed points in the environmental state where the configuration of the biota opposes changes in external perturbations to maintain the environmental state. This remains true even for a complex, high-dimensional environment (Dyke and Weaver, 2013). In this section, we thoroughly analyse the behaviour of coupled pairs of systems with respect to the value of the coupling parameter, D_E, which correlates the state of their environments. By doing so, we detail metrics to shed light on the coincidence of transitions in connected systems. We perturb one system from its initial stable state to an adjacent one (whose basin of attraction borders that of its initial state, if any). The behaviour of the model is trivial for the limiting cases of $D_E = 0$ and $D_E \to \infty$. In the



(a) Distance between adjacent fixed points with varyied coupling strength.

(b) Codistribution of changes in environmental state without cascades.

Figure 4: A system perturbed from the basin of attraction of a stable point undergoes a transition to the next stable fixed point. Fig. 4a shows how step sizes between fixed points are distributed in systems coupled to a stable environmental state. Fig. 4b shows, to first order, the feedback between connected systems; when a transition in one system does not coincide with a neighbour's transition, it produces only a small change in the environmental state inversely proportional to the gradient of $\boldsymbol{F}(\boldsymbol{E})$. Deviations from this covariate distribution are caused by cascading transitions.

first case, systems are uncoupled, while in the second case, the state of the environment is uniform across coupled systems, and the behaviour of the model is identical to a single system whose biotic effects consists of the sum of all systems' biota. In this case, correlations in changes between neighbouring systems is complete, as transitions cannot be confined to a single system.

To begin, we examine the probability distribution of transition sizes, $|\Delta\boldsymbol{E}|$ in a system coupled to another with an environmental state equal to its initial state. Previous work determined the expected number density of stable fixed points which can be used to find the mean transition size in an uncoupled system, given by Eq. (8). The simplest way to determine the form of this distribution however, is through Monte Carlo simulation whose results are given by Fig. 4a, which shows increasing coupling strength more strongly correlates environmental states in coupled systems, reducing the mean transition size slightly.

Next, we determine the response in the neighbouring system to the abrupt change in environmental diffusion which, to first-order (neglecting feedback), is inversely proportional to the gradient of its biotic effect function, $\boldsymbol{F}(\boldsymbol{E})$, in the direction of the change in environmental diffusion, $\Delta\boldsymbol{E}$. As the sum of the independent random contributions from the biota, the function is normally distributed with the variance fixed at unity by Eq. (6). The derivative $\nabla_{\Delta\boldsymbol{E}}\boldsymbol{F}(\boldsymbol{E})$ (abbreviated to $\boldsymbol{F}'(\boldsymbol{E})$) is trivially shown to be independent of $\boldsymbol{F}(\boldsymbol{E})$ and similarly distributed with variance $\frac{1}{2\sigma_E^2}$. In order to produce a fixed point, the function must cross the plane $\boldsymbol{F}(\boldsymbol{E}) = \boldsymbol{0}$, and therefore the likelihood of the function producing a fixed point in a small interval increases linearly with the magnitude of the gradient, and the gradient distribution becomes

$$\rho(|\boldsymbol{F}'|) \propto |\boldsymbol{F}'|^N \exp\left(-\sigma_E^2|\boldsymbol{F}'|^2\right) .\,. \tag{13}$$

To find the feedback from a transition in a neighbouring system, we require the reciprocal distribution which may be found from the moment generating function of the product $\frac{|\Delta\boldsymbol{E}|}{|\boldsymbol{F}'|}$. Having established the distribution of transition sizes through simulation, and the distribution of first-order feedbacks, we can provide the covariate distribution changes to the environmental state of two coupled systems, shown in Fig. 4b. Note that we use the notation $|\Delta\boldsymbol{E}_{1,2}|$ as shorthand for the two codependent variables $|\Delta\boldsymbol{E}_1|, |\Delta\boldsymbol{E}_2|$. Significant deviations from this distribution may be caused by coincident transitions between the coupled systems, providing us with a metric to evaluate the susceptibility of such coupled systems to cascading transitions. We now simulate connected systems with a range of connection strengths and compare the resulting codistribution of $\Delta\boldsymbol{E}_1$ to the non-transitioning probability density by taking the residual normalised by $\mathcal{E}$,

$$R^2 = \frac{1}{\mathcal{E}^2}\iint\Big(\rho(|\Delta\boldsymbol{E}_{1,2}|) - \rho_{\text{sim}}(|\Delta\boldsymbol{E}_{1,2}|)\Big)^2 \mathrm{d}|\Delta\boldsymbol{E}_{1,2}| \tag{14}$$

where ρ and ρ_{sim} are the probability distributions computed in Fig. 4b and generated by simulated systems respectively. Large values of R^2 indicate greater departure from

D_E	R^2	$\mathrm{corr}\big(\lvert\Delta \boldsymbol{E}_{1,2}\rvert\big)$	$\mathrm{corr}_{\mathrm{sim}}\big(\lvert\Delta \boldsymbol{E}_{1,2}\rvert\big)$
$\frac{1}{10}\mathcal{E}$	$\mathbf{0.003}$	0.12	$\mathbf{0.17}$
$\frac{1}{5}\mathcal{E}$	$\mathbf{0.013}$	0.13	$\mathbf{0.30}$
$\frac{1}{2}\mathcal{E}$	$\mathbf{0.080}$	0.16	$\mathbf{0.59}$
$\mathcal{E}$	$\mathbf{0.25}$	0.21	$\mathbf{0.67}$

Table 1: We measure changes to the environmental state of two coupled systems when one undergoes a transition into a new stable state. We quantify the discrepancy between the codistribution where there are no coincident transitions, where only first order coupling is considered, and a simulation which allows such events to occur.

the non-transitioning distribution and can be attributed to small higher-order feedback, and an increasing coincidence of transitions in connected systems. In addition to this, we can compare the correlation of $|\Delta \boldsymbol{E}_1|$ and $|\Delta \boldsymbol{E}_2|$ from the simulated systems to that predicted by Fig. 4b. A higher correlation indicates the state changes in connected systems have a greater degree of linear dependence, corresponding to increases in the coincidence of transitions.

Results

A comparison between the statistics of our non-transitioning model with first-order feedback and a simulated coupled system is provided in Table 1. Our analysis in the previous section has shown that, to first-order, the coupling between a transitioning system and its neighbour is related to the distribution of the effect function gradient. This gradient is responsible for the homeostatic properties of individual systems as it dramatically reduces the magnitude of changes to its state in response to changes in forcing from neighbouring systems or external sources; significant increases in these perturbations cause only marginal changes to the state of the model, providing the homeostatic fixed point behaviour prevails. This can be seen by examining the correlation coefficient for the non-transitioning system, $\mathrm{corr}\big(|\Delta \boldsymbol{E}_{1,2}|\big)$, which measures the linear dependance of changes to the state of the coupled systems. In contrast, our simulated system produces much larger values, $\mathrm{corr}_{\mathrm{sim}}\big(|\Delta \boldsymbol{E}_{1,2}|\big)$; the increase in linear dependence is therefore a result of coincidence of large transitions in the environmental state in the coupled system. This difference is mirrored by the integrated residuals, R^2, which quantifies the difference between the non-transitioning and simulated distribution of environmental changes.

Dyke and Weaver's (2013) simple ecosystem model exhibits a diversity of stable environmental configurations which emerge from a randomly parametrised biota. By connecting pairs of such systems, this paper has presented a minimal mechanism by which ecosystem-level transitions may cause cascades, which on a more extensive lattice of systems leads into large, potentially spanning transitions. Fig. 5 shows a lattice of one-thousand systems into a much larger metasystem. Changes in the coupling coefficient introduce longer range correlations in the environmental state and, when perturbed, lead to cascading transitions across parts the network.

The degree of connectivity, and the strength of connections between real Earth systems is unclear although these questions are ever more pressing; the environment and its biotic inhabitants are subjected to ever increasing anthropogenic pressures both locally, such as land use change, and globally, such as emissions of greenhouse gasses (Steffen et al., 2015). Investigating the robustness and limitations of regulatory mechanisms on Earth is as important as understanding the influence of a transition both in terms of alternative stable states, and the influence of such a dramatic change on coupled Earth systems (Williams and Lenton, 2010). This work is an early step towards understanding the collective behaviour of coupled multi-stable complex systems and lays the foundation for a more thorough and general analysis of correlations and transitions within the network.

Acknowledgements

This work was supported by a University of Southampton EPSRC Doctoral Training Partnership 2014 grant EP/M50662X/1.

References

Barnosky, A. D., Hadly, E. A., Bascompte, J., Berlow, E. L., Brown, J. H., Fortelius, M., Getz, W. M., Harte, J., Hastings, A., Marquet, P. A., Martinez, N. D., Mooers, A., Roopnarine, P., Vermeij, G., Williams, J. W., Gillespie, R., Kitzes, J., Marshall, C., Matzke, N., Mindell, D. P., Revilla, E., and Smith, A. B. (2012). Approaching a state shift in Earths biosphere.

Chavez, V. A., Doncaster, C. P., Dearing, J. A., Wang, R., Huang, J.-l., and Dyke, J. G. (2013). Detecting regime shifts in artificial ecosystems. In *Advances in Artificial Life, ECAL 2013*, pages 625–632.

Dyke, J. G. (2010). The Daisystat: A model to explore multidimensional homeostasis. In *Artificial Life XI, Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, pages 349–359. MIT Press, Cambridge MA.

Dyke, J. G. and Weaver, I. S. (2013). The emergence of environmental homeostasis in complex ecosystems. *PLoS computational biology*, 9(5):e1003050.

Goldblatt, C., Lenton, T. M., and Watson, A. J. (2006). Bistability of atmospheric oxygen and the Great Oxidation. *Nature*, 443:683–686.

Harvey, I. R. (2004). Homeostasis and rein control: From daisyworld to active perception. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, ALIFE*, volume 9, pages 309–314.

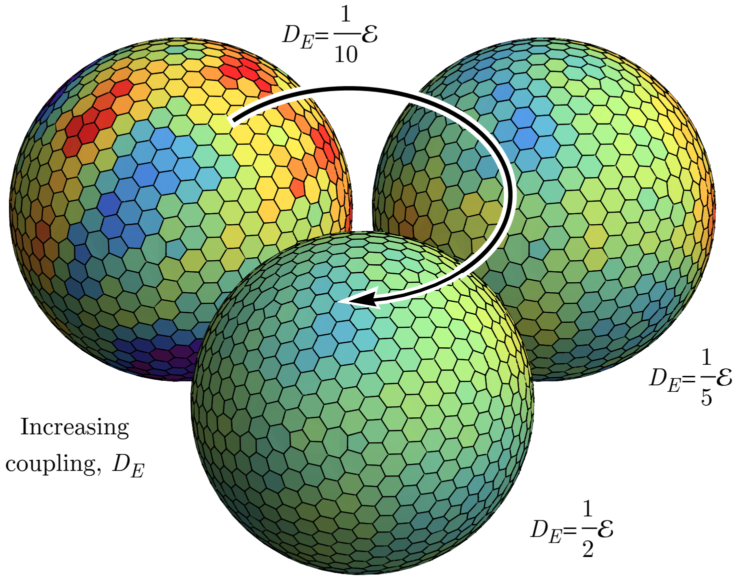

$D_E = \frac{1}{10}\mathcal{E}$
$D_E = \frac{1}{5}\mathcal{E}$
$D_E = \frac{1}{2}\mathcal{E}$
Increasing coupling, D_E

Figure 5: A lattice of connected systems mirrors the studied behaviour of connected pairs of systems; increasing coupling strength increases the correlation between the environmental state of groups of systems. Along with correlations comes the likelihood of local transitions causing cascades, dramatically changing the environmental state of large sections of the network.

Hoffman, P. F. and Schrag, D. P. (2002). The snowball Earth hypothesis: Testing the limits of global change. *Terra Nova*, 14(3):129–155.

Holme, P. and Newman, M. E. J. (2006). Nonequilibrium phase transition in the coevolution of networks and opinions. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 74(5).

Jones, C. G., Lawton, J. H., and Shachak, M. (1994). Organisms as ecosystem engineers. *Oikos*, 69:373–386.

Kleidon, A. (2011). How does the earth system generate and maintain thermodynamic disequilibrium and what does it imply for the future of the planet? *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 370:1012—-1040.

Lenton, T. M., Held, H., Kriegler, E., Hall, J. W., Lucht, W., Rahmstorf, S., and Schellnhuber, H. J. (2008). Tipping elements in the Earth's climate system. *Proceedings of the National Academy of Sciences of the United States of America*, 105(6):1786–1793.

Lenton, T. M. and Williams, H. T. P. (2013). On the origin of planetary-scale tipping points.

Odling-Smee, F. J., Laland, K. N., and Feldman, M. (2003). *Niche Construction: The Neglected Process in Evolution*. Princeton: Princeton University Press.

Punithan, D. and McKay, R. (2013). Collective Dynamics and Homeostatic Emergence in Complex Adaptive Ecosystem. In *Advances in Artificial Life, ECAL 2013*, pages 332–339.

Sanders, D. and van Veen, F. J. F. (2011). Ecosystem engineering and predation: The multi-trophic impact of two ant species. *Journal of Animal Ecology*, 80:569–576.

Stanley, H. E. (1987). Introduction to phase transitions and critical phenomena. *Introduction to Phase Transitions and Critical Phenomena, by H Eugene Stanley, pp. 336. Foreword by H Eugene Stanley. Oxford University Press, Jul 1987. ISBN-10: 0195053168. ISBN-13: 9780195053166*, 1.

Steffen, W., Richardson, K., Rockström, J., Cornell, S., Fetzer, I., Bennett, E., Biggs, R., Carpenter, S. R., de Wit, C. A., Folke, C., Mace, G., Persson, L. M., Veerabhadran, R., Reyers, B., and Sörlin, S. (2015). Planetary Boundaries: Guiding human development on a changing planet. *Science*, 347(6223).

Watson, A. J. and Lovelock, J. E. (1983). Biological homeostasis of the global environment: the parable of Daisyworld. *Tellus B*, 35B(4):284–289.

Weaver, I. S. and Dyke, J. G. (2012). A novel approach to analysing fixed points in complex systems. In Gilbert, T., Kirkilionis, M., and Nicolis, G., editors, *Proceedings of the European Conference on Complex Systems 2012*, pages 523–533. Springer.

Weaver, I. S. and Dyke, J. G. (2013). Tipping points in Complex Coupled Life-Environment Systems. In *Advances in Artificial Life, ECAL 2013*, pages 387–394. MIT Press.

Wilkinson, M. T., Richards, P. J., and Humphreys, G. S. (2009). Breaking ground: Pedological, geological, and ecological implications of soil bioturbation.

Williams, H. T. P. and Lenton, T. M. (2007). The Flask model: emergence of nutrient-recycling microbial ecosystems and their disruption by environment-altering rebel organisms. *Oikos*, 116(7):1087–1105.

Williams, H. T. P. and Lenton, T. M. (2010). Evolutionary regime shifts in simulated ecosystems. *Oikos*, 119(12):1887–1899.

Evolving an optimal group size in groups of prey under predation

Patrick B. Haley[1,3], Randal S. Olson[2,3], Fred C. Dyer[2,3], and Christoph Adami[2,3]

[1]The University of Texas at Austin, Austin, TX 78712
[2]Michigan State University, East Lansing, MI 48824
[3]BEACON Center for the Study of Evolution in Action, East Lansing, MI 48824
patrick.haley@utexas.edu, olsonran@msu.edu, fcdyer@msu.edu, adami@msu.edu

Considerable progress has been made in understanding the evolutionary forces underlying animal group-living behavior. Even so, Krause and Ruxton (2002) identified optimal group size as an under-researched area characterized by unwieldy large-predator study systems and simulations based on group rather than individual decision-making. Therefore, we present a simple, flexible simulation of foraging and predation that demonstrates that the evolution of an optimal, evolutionarily stable group size is in fact possible.

Prey genomes are evolved in subpopulations, each with a coevolved group size factor. These groups can be either heterogeneous or homogeneous. Genome fitness is determined by placing the subpopulation in the predator simulation used in Olson et al. (2014). Fitness-proportionate selection first acts on entire subpopulations (where a subpopulation's fitness is the mean fitness of its genomes) and then on genomes within each subpopulation (if the subpopulation is heterogeneous). Group size can also mutate between generations, with selection again choosing candidates for removal when a group shrinks and reproduction when it grows.

To study the potential disadvantages of living in large groups (e.g., competition for mates and resources), we apply a grouping penalty to foraging prey proportionate to the size of the group. This penalty is described by the equation:

$$\text{Food} = \frac{1}{G^P} \tag{1}$$

where G is the group size and P is the grouping penalty.

We find that at small grouping penalties, large group sizes evolve to share the expensive task of anti-predator vigilance. As the grouping penalty increases, group size declines gradually, causing individual vigilance to increase in turn. Once a large enough grouping penalty is reached, group-living ceases to be viable strategy, and prey instead evolve to live as solitary individuals.

When and how quickly this decline in group-living occurs is a function of reproductive strategy. Group-living falls off quickly in iteroparous prey, while group-living is preserved among semelparous prey until much larger grouping penalties are reached. In contrast, group composition has little impact on when group-living is no longer sustainable.

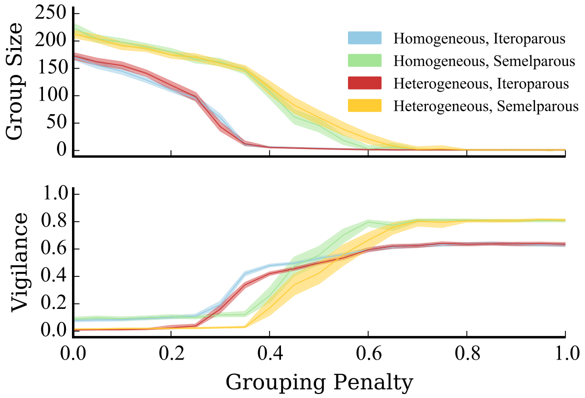


Figure 1: Group size and vigilance across various grouping penalties. Shaded regions show 95% confidence intervals.

These experiments represent an intial foray into using simulations to understand whether an optimal group size exists. An advantage of our system is that in the future we can model the individual decision-making endorsed by Krause and Ruxton (2002). Future simulations also can consider complex factors like food scarcity. Still, these results suggest that prey can evolve to live at an optimal, stable group size, which is mediated by the costs of living in groups.

Acknowledgements

This research has been supported in part by the National Science Foundation (NSF) BEACON Center under Cooperative Agreement DBI-0939454. We thank the Michigan State University High Performance Computing Center and the Institute for Cyber-Enabled Research (iCER).

References

Krause, J. and Ruxton, G. D. (2002). *Living in Groups*. Oxford University Press, Oxford.

Olson, R. S., Haley, P. B., Dyer, F. C., and Adami, C. (2014). Exploring the evolution of a trade-off between vigilance and foraging in group-living organisms. *arXiv preprint arXiv:1408.1906*.

A Metamodel for the Evolution of Evolution

Paul Andrews and Susan Stepney

Department of Computer Science, University of York, UK
susan.stepney@york.ac.uk

Abstract

The ability of evolution to influence its own course in microorganisms such as bacteria is a desirable property for computational systems. As a step towards exploiting this, we define a metamodel for the Evolution of Evolution. The metamodel is based on the concepts of structure and process, which are embodied together as the Machine. By describing different types Machines and structures we can capture a flexible metamodel to form the underpinnings for a new generation of evolutionary algorithms.

Introduction

The concept of *Evolution of Evolution* (EvoEvo) stems from the idea that evolution within organisms is able to influence its own course, and is derived from the observation that the molecular systems involved in evolutionary processes have themselves resulted from past evolution (Beslon et al., 2014). This has resulted in phenomena such as the evolution of robustness, mutation operators, mutation rates, and evolvability. Given the timescale of evolution, EvoEvo phenomena are most evident in bacteria and viruses, which can adapt rapidly and efficiently to changing environments by accelerating their evolution.

Inspired by the evolutionary dynamics observed in bacteria and viruses, the EU funded FP7 project, EvoEvo[1], aims to harness EvoEvo phenomena to create evolvable software systems. Specifically, the aim is to develop new algorithmic approaches to address dynamically changing *open-ended* engineering problems in which solutions adapt to previously unknown conditions and potentially evolve new types of solutions. To capture the relevant EvoEvo phenomena, the evolutionary dynamics of experimental bacterial and viral systems are being studied to inform computational models and simulations (based on Knibbe et al. (2007); Crombach and Hogeweg (2007)). In turn, these models form the inspiration for a *computational framework* that captures EvoEvo processes and can be instantiated as evolutionary algorithms for a given engineering problem. It is the design of this framework that is the basis for this paper.

We present here a *metamodel* for EvoEvo that encapsulates and abstracts the relevant analogies of biological EvoEvo processes, specifically those observed in bacteria. Whilst a model of a system provides an abstract language for relevant concepts, a metamodel provides the language for writing a model by specifying the *kinds of things* that might be present in the model (Kleppe et al., 2003, Chapter 8). In Andrews et al. (2011) we discuss in depth the concept of metamodels in the context of modelling complex systems and developing bio-inspired engineering systems. A useful example of a metamodel is given in that paper: a metamodel of agent-based modelling might include concepts of Agent, Rule and Emergent. Based on that metamodel, a model of ant pheromone trails would then include an instance of Agent, the Ant, and an instance of Emergent, the Trail.

So, different, separate models that explore EvoEvo concepts can be instances of the same metamodel, and this EvoEvo metamodel can establish the core components of the aforementioned EvoEvo computational framework. For this framework to be successfully used by third parties to instantiate their own EvoEvo-based algorithms, having the engineering rigour of an explicitly defined metamodel is a must.

It is worth noting that there are any number of different EvoEvo metamodels that could defined. The approach taken here is based on the requirement that the metamodel not only provides a language that can capture EvoEvo concepts, but has a natural analogy to computation so that it can be easily translated into the eventual computational framework. Before introducing the full EvoEvo metamodel, we first describe a more general Machine metamodel that fulfils this computational analogy requirement, and therefore forms the building blocks of the EvoEvo metamodel itself.

Machine Metamodel

At the most abstract level, any model of a biological system or concept such as EvoEvo can be considered in terms of structures (e.g. DNA) and processes (e.g. evolution via natural selection) that describe how these structures change through time and space. Further, we introduce the notion

[1] http://evoevo.liris.cnrs.fr/

of a structure that implements a process (e.g. an enzyme). Here, we call this reified processes a *Machine*.

The Machine abstraction, informally conceptualised in Figure 1, provides us with a flexible language that is used to define our EvoEvo metamodel. A Machine is structure that implements a process that receives as input structures and energy, and returns as output (potentially) modified structures and (abstract) energy. Figure 1 also shows how Machines can form networks where inputs to one Machine are outputs from another. This allows us to compose a model of a system in which structures are subject to continual change via any number of processes that are driven by energy.

More formally, we can capture the Machine concepts using a class diagram, shown in Figure 2. This describes a Machine in terms of its relationship to Structure, Process, Symbol, and Energy.

Structure: composed from an ordering of Symbols. Does not itself posses internal behaviour or state (other than its own existence).

Symbol: a member of a given Alphabet that forms the building block of a Structure.

Alphabet: a set of possible Symbols. All Symbols that make up a Structure will be from the same Alphabet.

Process: acts upon Structures, potentially transforming them. Driven by Energy.

Energy: a quantity that drives the operation of Processes.

Machine: inherits from Structure and Process and so is an instance of both; a concrete Structure that reifies and implements a Process. As a Process, it can act upon other instances of Structure. It can store Energy and contain state.

There are three components to the Machine. First, as we have seen, the Machine extends the concept of Process, which means it exhibits some kind of behaviour. This behaviour can modify Structures that are the inputs to (and subsequent outputs of) the Machine. Second, a Machine can have state, which provides a memory to the Machine. One consequence of this is the ability to store parameters that shape the dynamics of the Machine's behaviour. Lastly, the Machine can (but is not required to) store energy that is used to drive the Machine's behaviour. In the absence of stored energy, the Machine's energy input must be sufficient to drive its behaviour. The Machine is not permitted to modify its own behaviour (Machines do not self-modify), but they can update their state.

The Machine also extends the concept of Structure. Hence a Machine (as Structure) can be passed to another Machine (as Process), allowing for its modification. But what does it mean for a machine to be data? Take for example an enzyme. The enzyme has a behaviour (catalyses a reaction) that can modify metabolites (structures). The enzyme itself, however, could be considered a Structure that was the output of a chemical reaction (Machine) that created it. Hence the same enzyme is either a Structure or Process *depending on the context*.

We can form an aggregate machine if the Structure/s out-

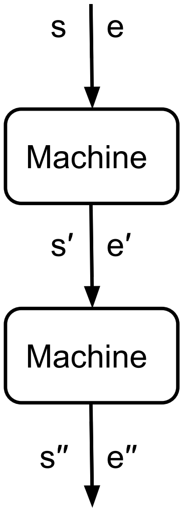


Figure 1: Machine takes structure, **s**, and energy, **e**, as inputs and returns potentially modified structures, **s**′, and energy, **e**′, as output. Networks of machines can form where outputs from one Machine are the inputs to the next.

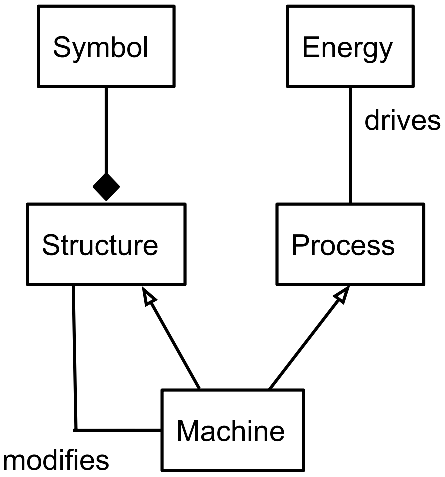


Figure 2: Class diagram showing the relationship between Machine, Structure, Process, Symbol, and Energy. Machine inherits from both Process and Structure, and modifies instances of Structure. Process is driven by Energy. Structures are composed of Symbols from an Alphabet (not shown).

put from one machine match the input expected for another (and the energy outputs and inputs are also satisfied). Figure 3 demonstrates the case for two machines M1 and M2 that can also be viewed as a single machine M3. Depending on the model that implements the Machine metamodel concepts, the ability to aggregate at some level of abstraction could be useful. For example a metabolic pathway could be viewed both as a series of enzyme machines or a single pathway machine. How this aggregation is handled will be specific to the implementing model, however it is not inconceivable that some kind of observer process (machine) would be involved.

Energy, Space and Time

Energy in the Machine metamodel draws heavily from that given by Hoverd and Stepney (2011). Energy is a combined generalised resource flux (e.g. think nutrients and sunlight). Importantly it is a limited resource, and the flux can be used, stored or simply ignored in which case it dissipates and it not used. There are three parts to the energy metatmodel:

Flux: represents a flow of energy from outside the modelled system. It can have a particular temporal pattern e.g. high level during day, but lower level during night.

Store: represents the component's ability to store energy.

Demand: represents a demand for an amount of energy from a model component.

A Machine can be both a Store and Demand for energy.

The metamodel also contains the concept of Space. Spaces are containers within which Machines and other Structures are located. Importantly, a Space provides a locality for these Machines and Structures, which defines the Machines' connectivity. For example the Space might define distances to other Machines/Structures or filters that describe what other Machines/Structures can be seen and interacted with. In addition, Time defines how this ordering changes. Thus together Space and Time determine how machines interact (when and with what), which, along with the machine behaviour, gives the machine network dynamic

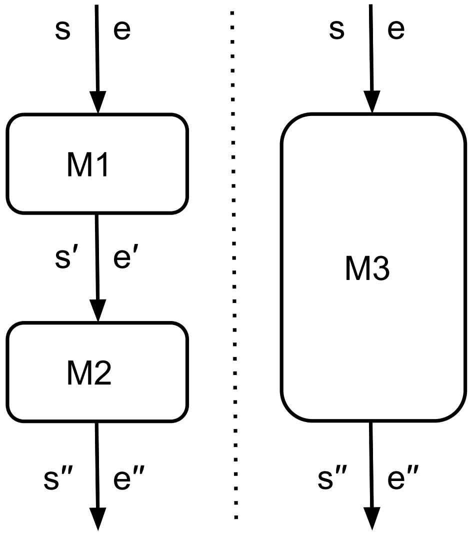


Figure 3: Machine M3 can be viewed as an aggregate of M1 and M2.

Creating Structures and Machines

We can extend the metamodel summarised above to include the ability to create and change Structures and Machines by introducing a MachineTemplate (MT) and Location Structure and three specific classes of Machine: Copier, Locator, and Constructor. These are shown in Figure 4 and summarised as:

MachineTemplate: a Structure containing the instructions for building a particular Machine, specifically: its initial state (e.g. the default parameter settings); components of the behaviour including what input structures it can receive, and what outputs structures are generated; energy storage ability. Importantly, MTs exist separately from machines and a Machine does not store its describing MT; they are different entities in the metamodel.

Copier: creates a copy of an input Structure, leaving the original Structure unchanged. The copied Structure could be exact (same Symbols), erroneous (potentially different Symbols from same Alphabet), or a translated copy (different Symbols from different Alphabet) based on the Copier's behaviour.

Locator: locates a sub-Structure of a Structure given Location identifiers that denote the beginning and end of the sub-Structure.

Location: a Structure that acts as an identifier on another Structure. Two specific Locations are required by the Locator, the Begin (cf. DNA promotor, start codon) and the End (cf. DNA terminator, stop codon).

Constructor: constructs a Machine from a MachineTemplate. It utilises a Copier and Locator Machine to construct the Machine's Structure representation from the MachineTemplate (cf. creating a polypeptide from RNA). The Machine's Structure is then given the property of Process, however that is defined in the model (cf. folding a polypeptide into its functional protein form).

It is interesting to note that different ConstructorMachines could create different Machines from the same MachineTemplates as they interpret the Symbols differently. Additionally, as the ConstructorMachine itself can have an associated MT, the Copier could potentially change the MTs for new ConstructorMachines, thus the system itself could dynamically modify how future Machines are interpreted.

With both Machine and MachineTemplate we have a system in which we can potentially change only instances of Machines – a Machine can modify an instance of another Machine – or all future copies of a given type of Machine by modifying its associated MachineTemplate.

The machine metamodel should be general yet expressive enough to model many different systems, not just the intended EvoEvo metamodel described in the rest of this paper. For example it can capture a very general model of computation: a Machine is an executable computing process (compiled or interpreted); MachineTemplate is the source code for a Machine; the ConstructorMachine is an interpreter/compiler; Space and Time are a process scheduler.

Requirements for EvoEvo

As discussed in the Introduction, EvoEvo is based on the idea that evolution is able to shape its own path given that the molecular systems involved in the evolutionary process have resulted from past evolution. Specifically, there are four characteristics of the genotype-to-phenotype mapping that can enable EvoEvo Beslon et al. (2014):

Variability: the ability to generate new phenotypes via changes in mutation rates and operators

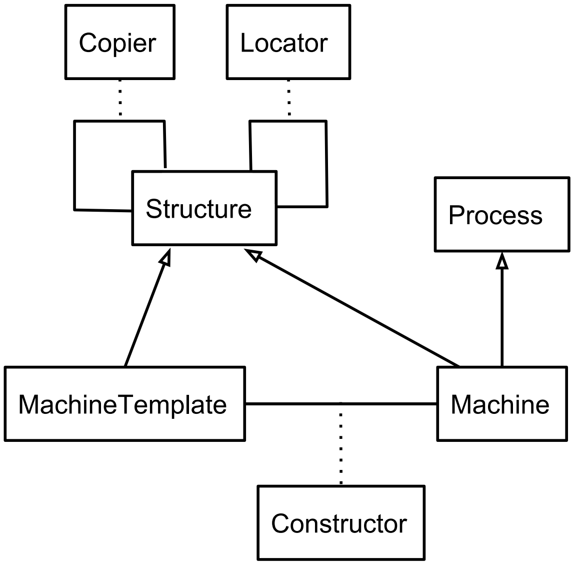


Figure 4: Class diagram showing the relationship between Structure, Machine, Copier, Locator, MachineTemplate, and Constructor. Copier and Locator inherit from Machine (not shown) and operate on Structures. A MachineTemplate is a Structure. A Constructor inherits from Machine (not shown) and creates a Machine from MachineTemplate.

Robustness: the ability to cope with mutational events without negatively impacting fitness

Evolvability: the ability to increase the proportion of mutational events that are favourable

Open-endedness: the ability to create new evolutionary avenues and targets.

These four characteristics emerge from the underlying processes and structures and the continual evolution of these processes and structures.

EvoEvo is concerned with allowing the genotype-to-phenotype mapping and the fitness landscape to evolve over time via indirect selection, leading to properties that enable evolution in dynamic environments Beslon et al. (2014). Targets for indirect selection are focussed on the organism's:

Genetic structures: numbers of genes, position of genes on genome, operons, non-coding sequences.

Networks: gene regulatory, metabolic and "social" (interactions between individuals)

These targets will become the focus for the EvoEvo metamodel that follows.

EvoEvo Metamodel

Here we extend the Machine metamodel to create a metamodel for EvoEvo that captures the desired EvoEvo concepts just described. This EvoEvo metamodel expresses the kinds of things we would expect to see in an EvoEvo model. This metamodel is based on the processes observed in bacteria, and has been inspired by studying the EvoEvo models of our project partners Knibbe et al. (2007); Crombach and Hogeweg (2007), as well as requirements for evolutionary computing. In brief, it introduces two types of Space, Individual and Environment, and a number of specialised Structures and Machines. Importantly we make a distinction between different types of machine that operate on different aspects of the Individual. First we describe the EvoEvo Spaces and Structures, followed by the EvoEvo Machines.

Spaces and Structures

We use the Space component from the Machine metamodel to represent both an Individual – such as a bacterium – and the concept of an Environment in which Individuals exists. Individuals and Environments contain various specialised Structures and Machines.

Individual A Space that represent an organism within (or potential "solution" to) an Environment. As a Space, the Individual contains any number of Machine instances, which interact with each other to form networks of the four types of Machine (see 'Machine Types' below). An Individual has a Genome consisting of at least one MachineRepository, which encodes MachineTemplates. A Phenotype for the Individual results from the combined action of its Machines with reference to its Environment. Based on the Phenotype, the Individual is the unit of Selection – the thing that is selected for within the Environment – and undergoes reproduction. The same individual will have different Phenotypes for different Environments. We further explore Phenotypes, Fitness and Selection later in the paper.

Environment A Space containing one or more Individuals as well as other Machines and Structures. As a Space, the Environment provides an ordering for its contents and thus describes the connectivity between Individual instances. The embodiment of an Individual in the Environment defines the "problem" that is being addressed by the Individual. For example, in a model of a bacterium this would be its ability to survive and reproduce, whereas for an evolutionary algorithm this would be a potential solution to an optimisation function. The Environment can be dynamic (Machines, Structures and Individuals can change over time), therefore this "problem" can also be dynamic. The Individual receives inputs from Environment in the form of Structures, such as Metabolites and Machines. The Environment can therefore be a source of epigenetic effects on the Individual, with these inputs interacting with the Machines within the Individual – such as those that operate on the Genome – which can in turn influence Machine expression.

Genome and MachineRepository The Genome is the source of heredity between an Individual and its offspring and is copied, and potentially, mutated during reproduction. The Genome consists of one or more MachineRepository Structures (cf. a chromosome). A MachineRepository is the source of MachineTemplates that encode the Individual's Machines (cf a gene). A MachineRepository is constructed from any number of Symbols from a single alphabet, and therefore stores any number of related MachineTemplates. Not all Symbols have to form part of a MachineTemplate (non-coding regions). Different MachineRepositories could be constructed from different Symbol alphabets for MachineTemplates for different types of Machine (see 'Machine Types' below). MachineRepositories act as a persistent store for MachineTemplates during an Individual's existence as well as allowing new MachineTemplates to evolve and for organisational pattern of MachineTemplates to occur between generations of Individuals. The presence of Begin and End Locations on a MachineRepository define location of a TranscriptionUnit. A set of Genome Machines (described below) operate on the MachineRepositories to carry out copying, mutations and transcriptions.

TranscriptionUnit A Structure that can be transcribed (copied) from a MachineRepository, and may contain any number of MachineTemplates. The section of MachineRepository that is subject to the copy is defined by the position of Locations (see below) and is the biological analogy of an operon, whilst the actual TranscriptionUnit Structure is analogous to mRNA. The TranscriptionUnit provides a mechanism to group related MachineTemplates so that the subsequent Machines are constructed together. It is noted that many evolutionary algorithms omit the the TranscriptionUnit concept, translating genes (MachineTemplates) straight from the Genome. Begin and End Locations on TranscriptionUnit define location of a MachineTemplate.

Metabolite A Structure that forms the basis for operations of the Metabolite Machines (see below). These might be the building blocks for an artifical chemistry or other symbols for processing in an evolutionary algorithm.

Machine Types

The targets for indirect selection identified above in 'Requirements for EvoEvo' – genetic structures and the gene regulatory, metabolic and social networks – provide us with a useful categorisation for machines in the EvoEvo metamodel. These machine types are:

Genome Machines: operate on the Genome, responsible for constructing Machines and reproduction.

Metabolism Machines: provide a potential "solution" to the "problem" that is being addressed by the Individual (e.g. staying alive in order to reproduce in the case of a bacterium).

Regulation Machines: help control the dynamics of Machine construction from the Genome by repressing or inducing the action of other genome machines.

Boundary Machines: control the transport of Structures to and from Individuals and its external Environment.

Instances of each machine type interact (via Structures) to form machine networks. Interactions also occur between these networks to express the Individual's Phenotype. There is a clear analogy to the work of Lones et al. (2013) who come to a similar categorisation of biochemical networks.

Genome Machines

The Genome Machines exist within an Individual and operate on its Genome and TranscriptionUnit Structures. Essentially, these are the Machines that both decode the information stored on these Structures creating new Machine instances, and the Machines that create copies of these Structures.There are 5 specific types of Genome Machine that would be relevant to all EvoEvo models: Transcriber, Translator, Expresser, Cloner and Reproducer.

Transcriber This Machine is responsible for producing TranscriptionUnits from a MachineRepository. It receives as input a Structure (such as a MachineRepository) and two

Locations, Begin and End, which define the beginning and end locations of the DataStruture to be copied (i.e. the location of the operon). As output, the Transcriber returns the original Structure unchanged and the copy that has been made. Figure 5 demonstrates the Transcriber in action. We can define the process implemented by the Transcriber in terms of the Locate and Copy machines described in the machine metamodel. Locate is used twice to find the Begin and End Locations and the Copy is used once. In the case of a real biological system, the Copy will slightly change Symbols from the DNA bases (C, G, A and T) to RNA bases (C, G, A and U).

Translator This Machine is responsible for producing Machines from TranscriptionUnits. Similar to the Transcriber, it receives as input a Structure (the TranscriptionUnit) and two Locations, Begin and End, which signal the beginning and end locations of MachineTemplates within the TranscriptionUnit. As output, the Translator returns the original TranscriptionUnit and a Machine for every MachineTemplate located. Figure 5 demonstrates the Translator in action. We can define the process implemented by the Translator in terms of the Locate, Copy and Constructor machines described in the machine metamodel. For each MachineTemplate, Locate is used twice to find the Begin and End Locations, the Copy is used once to build the Structure representation of the Machine, and Constructor is used to turn this Structure into a Process (and hence an instance of Machine).

Expresser This Machine operates on a MachineRepository controlling the expression of TranscriptionUnits. It operates by providing the Transcriber with the Locations and MachineRepository at the operon chosen for transcription. By working in conjunction with regulation machines (Regulators) this allows for different expression rates for TranscriptionUnits.

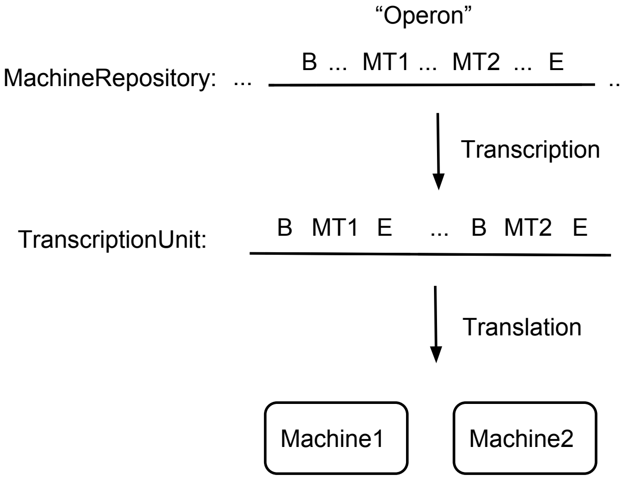


Figure 5: The interaction between the MachineRepositor, TranscriptionUnit, MachineTemplates, Machines, Transcription and Translation. B denotes Begin, E denotes End.

Cloner The Cloner is a Copier Machine that performs an inaccurate copy of a MachineRepository. This provides a mechanism to introduce mutations and recombinations to a copy of the MachineRepository. It takes as input a MachineRepository and returns as output the original MachineRepository (unchanged) and inaccurate copy of MachineRepository. How the Cloner changes the copy is problem specific – it will contain rules on which data symbols can change and how they change. Different Cloners will introduce different types of mutation and recombination.

Reproducer Responsible for creating a new Individual based on the current Individual. It requires each MachineRepository to be copied by the Cloner, with the child Individual receiving the cloned MachineRepositories, potentially generating novelty between generations of Individual. The child Individual will also inherit a share of the Structures and Machines present in the parent. Depending on the model that instantiates the Reproduce, one option for the parent Individual is that it dies (see 'Structure Degradation and Death') as a consequence of the Reproducer.

Metabolism Machines

Metabolism is the set of processes within an Individual that maintain its *viability* as an Individial within an Environment. For an organism such as a bacterium, this would be the staying alive in order to reproduce, whilst for an evolutionary algorithm this would be equivalent to the provision of a fitness function. Essentially, metabolism provides a potential "solution" the "problem" that is being addressed by the Individual (e.g. in the case of a bacterium).

Metabolisers are machines that perform the metabolism of the Individual, which together form the metabolism network. The behaviours implemented by the Metabolisers will be very much specific to the model that instantiates the metamodel. For example a model of biology might have an set of enzyme-related reactions, whilst for an evolutionary algorithm the Metabolisers they will collectively form the fitness function. As the definition of metabolism is specific to the model that instantiates the metamodel, there is little more we can say about specific Metabolisers.

The metabolic network will interact with the other machine networks, which together gives the Individual its Phenotype. The role fulfilled by the Metabolisers is to determine the viability of the individual in the current Environment. This viability is a key component in establishing fitness for the selection dynamics (see 'Phenotype, Fitness and Selection'). How selection is performed will be model specific.

Regulation Machines

Regulation within an Individual is performed by Regulator Machines. Specifically, these are a network of Machines that provide a set of extra dynamics on top of the genome machine network, controlling the action of the Transcriber and Translator. Both the MachineRepository and TranscriptionUnit are open to the action of the Regulators, cf. transcription factors and regulatory RNA.

Regulators identify their units of regulation (e.g. operon) by locating an associated Begin Location and BindingSite Location that is unique to the Regulator. The Regulator will then interact with the Machine responsible for coping that unit of regulation (e.g. Transcriber or Translator) to either repress or induce its action, which will either down- or up-regulate its expression.

The combined action of the Regulators forms networks such as gene-regulatory networks. Interactions can also occur with metabolism and boundary machines providing feedback control so that Machine expression can adapt to environmental pressures and signals. Additionally, the Regulators are encoded on the MachineRepository so the regulatory networks can themselves evolve over the generations.

Boundary Machines

Boundary Machines control the interface between the Individual and the Environment controlling transport across that boundary, determining what gets in and what goes out. If the Boundary Machines are present on the MachineRepository, they will be able to evolve the dynamics of this transport. In controlling the boundary, these Machines provide the environmental/epigenetic context for the Individual.

Any Structure is capable of being transported in and out of the Individual. Three specific types would be Metabolites, Metaboliser Machines and MachineRepositories. The first two are provide a mechanism for resources to flow in and out of the Individual, whilst the later enables processes such as horizontal gene transfer. In bacteria, horizontal gene transfer is takes place via plasmids, which are akin to MachineRepositories in this metamodel.

Structure Degradation and Death

The degradation of Structures (and by definition, Machines) drives change as new copies of Machine will be needed to replace those that degrade. Degradation is captured by the Entropy Machine/s, which are defined at the level of the Environment. Different Entropy Machines may be required for the different types of Machine. Importantly, Entropy Machines should never itself be subject to evolutionary change as they are essentially defining the *physics* of the system in which Individuals are evolving. For example, an Individual should not be able to cheat death by redefining the 2nd law of thermodynamics.

Entropy Machines degrade Structures into their constituent Symbols, which are then available to the containing space (Individual or Environment). Rates of degradation can be different for different Structures (e.g. DNA is more stable than RNA). The degradation behaviour within an Individual can lead to death – insufficient Machines and Structures to remain viable. A dead Individual can be release its contents into the Environment.

Phenotype, Fitness and Selection

We established above that the Indvidual's Metabolisers determine its *viability*. The reproduction of an Individual occurs as a result of this viability and the current conditions in the Environment, which is determined by the Selection machine. Selection does not just select the most viable (the fittest) Individuals to reproduce, but takes into consideration conditions in the environment. So, the Selection Machine will reproduce an individual based on a function of the Individual, e.g. does it have enough resource, and the Environment, e.g. is there enough space or is there a suitable mate (if the model incorporates sexula reproduction). Like the Entropy Machine, the Selection Machine is defined as a property of the system does not change behaviour over the shorter timescales of any simulation that implements the metamodel.

The timing of the Selection Machine is defined as part of the model that applies the metamodel. Selection could apply to all Individuals at the same time (cf evolutionary computing) or as and when certain conditions are met (cf bacteria when it has aquired sufficient resources and space).

Discussion

Embodiment and Machine Origins

Other than the requirement for fixed Entropy and Selection Machines, the metamodel says little about where Machines and MachineTemplates arise. It will be a modelling/design decision as to which MachineTemplates are located on the MachineRepository (and so the associated Machines are expressed by the Individual's genome machinery), or are statically defined ("hard-coded") into the system. These statically defined Machines constitute the physics of the system – those parts that cannot change. In the EvoEvo framework, the computational problem will dictate which machines are fixed and which are able to evolve. For each problem, a suitable set of Machines will need to be designed.

MachineTemplates on the MachineRepository are obviously evolvable within the system. This *embodiment* of MachineTemplates allows the instructions on how to make the functional parts of the system (the Machines) part of the system itself. This makes them accessible to change, and potentially would allow the system can change its own encoding in an *open-ended* way.

Representations

The flexibility of the Machines upon which the EvoEvo metamodel is based has a number of consequences for logic and data representations. The Machines essentially apply the semantics to the syntax given by the Structures. Likewise meaning is only given to the Symbols on a Structure, such as MachineTemplate or MachineRepository, when it is processed by a Machine. It is the Machines present in the system that define the overall behaviour, and the expression of these Machines can be controlled by the system itself.

The same MachineTemplate can be interpreted in different ways by different Machines to, in principle, construct entirely different Machines. Similarly meaning is only given to other data representations such as orderings on a Structure (e.g. operons and the positions of different MachineTemplates) when they are processed by other Machines.

Different Begin Locations on the same TranscriptionUnit can be used as targets for different Translator machines. For example one Begin Location could be target for the Translator of Metabolite Machines, whilst a different Begin Location can be used for a Translator of Genome Machines. This would allow the two different Translators to build different Machines types from the same MachineRepository

Within the genomes of organisms seen on Earth there are different levels of structure and organisation such as: base, codon, gene, operon. The ability to define the behaviour of Machines that operate on a MachineRepository, and to allow these Machines to evolve, could allow different representations of the MachineRepository to evolve. This would give us insight about evolution *as it could be* in an artificial world.

With regard to computational evolutionary systems, being able to move between different representations of the same MachineRepository would help improve performance and exploit the best search space dynamics (exploration versus exploitation). For example some Copiers might mutate a level equivalent to bases (A,C,G,T or binary strings), whilst others could mutate at the level of MachineTemplates (e.g. moving, duplications). Which Copiers are currently used would be defined by the system itself if their MachineTemplates are evolvable.

Conclusions and Future Work

We have outlined and discussed above a metamodel for EvoEvo, which aims to abstract and interpret observed biological concepts in a form suitable for in silico implementation. In particular, the inspiring biological concepts arise from the evolutionary dynamics observed in bacteria and viruses. The metamodel is the first step towards building suitable computational analogues of the EvoEvo mechanisms that will form the basis of a computational framework enabling the development of novel evolutionary engineered systems.

The Machine metamodel gives us the language of process and structure, and the Machine which embodies process within a structure. The Machine metamodel also provides descriptions of Machines that enable us to create Machines encoded within MachineTemplate structures. The generalised Machine metamodel then provides the language to specify the EvoEvo concepts we obseve in the inspiring bacterial systems. We noted in the Introduction that there are any number of different EvoEvo metamodels that could defined. The concepts presented here in this metamodel have been specifically selected to enable the next stage of our research: to implement these concepts *in silico*.

Having established the EvoEvo metamodel, we plan to show how it can capture the models developed by our project partners, including Knibbe et al. (2007); Crombach and Hogeweg (2007), as well as being a model for evolutionary algorithms. Given this, we can address our ultimate goal of to implement the EvoEvo computational framework that will allows the user to create problem-solving algorithms based on the EvoEvo concepts in the metamodel.

As highlighted above, it is no coincidence that the Machine metamodel has many natural analogies to computational systems as it was a requirement of the metamodel to provide the basis for the implementation of the computational framework. One desirable analogy is the natural link between Machines and self-contained objects, which provides a natural link to a number of concurrency techniques. Our hope is implement a powerful EvoEvo framework, both in term of dynamic and computational power.

Acknowledgements

This work was funded by the EU FP7 project EvoEvo, grant number 610427. Thanks to Simon Hickinbotham and Chris Timperley for their useful discussions.

References

Andrews, P. S., Stepney, S., Hoverd, T., Polack, F. A. C., Sampson, A. T., and Timmis, J. (2011). CoSMoS process, models, and metamodels. In Stepney, S., Welch, P., Andrews, P. S., and Ritson, C. G., editors, *Proceedings of the 2011 Workshop on Complex Systems Modelling and Simulation, Paris, France, August 2011*, pages 1–13. Luniver Press.

Beslon, G., Elena, S. F., Hogeweg, P., Schneider, D., and Stepney, S. (2014). Evolution of evolution: Description of work. http://evoevo.liris.cnrs.fr/description-of-the-evoevo-project/.

Crombach, A. and Hogeweg, P. (2007). Chromosome rearrangements and the evolution of genome structuring and adaptability. *Molecular biology and evolution*, 24(5):1130–1139.

Hoverd, T. and Stepney, S. (2011). Energy as a driver of diversity in open-ended evolution. In *ECAL 2011, Paris, France, August 2011*, pages 356–363. MIT Press.

Kleppe, A., Warmer, J., and Bast, W. (2003). *MDA Explained: the Model Driven Architecture: practice and promise.* Addison-Wesley.

Knibbe, C., Coulon, A., Mazet, O., Fayard, J.-M., and Beslon, G. (2007). A long-term evolutionary pressure on the amount of noncoding DNA. *Mol. Biol. Evol.*, 24(10):2344–2353.

Lones, M. A., Turner, A. P., Fuente, L. A., Stepney, S., Caves, L. S. D., and Tyrrell, A. M. (2013). Biochemical connectionism. *Natural Computing*, 12:453–472.

Informational parasites in code evolution

Andrés C. Burgos[1] and Daniel Polani[2]

[1,2]Adaptive Systems Research Group, University of Hertfordshire, Hatfield, UK
[1]a.c.burgos@herts.ac.uk

Abstract

In a previous study, we considered an information-theoretic model of code evolution. In this model, agents bet on (common) environmental conditions using their sensory information as well as that obtained from messages of other agents, which is determined by an interaction probability (the structure of the population). For an agent to understand another agent's messages, the former must either know the identity of the latter, or the code producing the messages must be universally interpretable.

A universal code, however, introduces a vulnerability: a parasitic entity can take advantage of it. Here, we investigate this problem. In our specific setting, we consider a parasite to be an agent that tries to inflict as much damage as possible in the mutual understanding of the population (*i.e.* the parasite acts as a disinformation agent). We show that, after introducing a parasite in the population, the former adopts a code such that it captures the information about the environment that is missing in the population. Such an agent would be of great value, but only if the rest of the population can understand its messages. However, it is of little use here, since the parasite utilises the most common messages in the population to express different concepts.

Now we let the population respond by updating their codes such that, in this arms race, they again maximise their mutual understanding. As a result, there is a code drift in the population where the utilisation of the messages of the parasite is avoided. A consequence of this is that the information that the parasite possesses but which the agents lack becomes understandable and readily available.

Introduction

Codes shared among entities are ubiquitous in nature, not only present in biological systems, but also, at the least, in technological ones (Doyle, 2010). We define a code as a probabilistic mapping from an "input" random variable (*e.g.* environmental variable) to a set of outputs (*e.g.* messages). A code, then, implies a representation of the input variable. When representations are shared among entities, they become conventions which are used for communication (Burgos and Polani, 2014, 2015). The correct use of these conventions for communicating can be interpreted as "honest signalling". For instance, the TCP/IP protocol allows the interaction of hardware and software in a code-based, "plug-and-play" fashion, as long as they obey the protocol (Doyle, 2010). In biology, the genetic code acts as an innovation-sharing protocol, one that allows the exchange of innovations through horizontal gene transfer (HGT) (Woese, 2004).

However, communication protocols introduce vulnerabilities: parasitic agents can take advantage of them (Ackley and Littman, 1994; Doyle, 2010). For instance, the chemical cues that ant colonies use to recognise nest-mates can be mimicked by slave-making workers for social integration (D'Ettorre et al., 2002). On the Internet, one can take advantage of machine communication protocols (TCP/IP) to force target computers to perform computations on behalf of a remote node, in what is called "parasitic computing" (Barabási et al., 2001).

Parasites benefit from their interaction with other agents, while reducing the fitness of the attacked hosts. Nevertheless, parasites can be a positive force in evolution. For instance, they can be generators of biodiversity, achieving more resistance to future attacks (Brockhurst et al., 2004). In an artificial setting, the presence of parasites was shown to attain more efficient communication between agents of a population, increasing their reproductive success (Robbins, 1994). Furthermore, some authors suggest that a healthy ecosystem is one rich in parasites (Hudson et al., 2006). In this work, we study this apparent contradiction from an information-theoretic perspective.

We look at some aspects of the co-evolutionary arms race between host and parasite. Particularly, we would like to characterise informationally the behaviour of parasites and the consequences for the host. For this purpose, we assume a simple scenario where organisms seek to maximise their long-term growth rate by following a bet-hedging strategy (Seger and Brockmann, 1987). We know that maximising their information about the environment achieves this (Shannon, 1948; Kelly, 1956). Then, individuals obtaining extra environmental information from other individuals will have an advantage over those that do not, since they would be able to better predict the future environmental conditions (Donaldson-Matasci et al., 2010). However, as we showed in a previous work, for simple agents which do not have the ability to identify who they are listening to, a shared code among the population is necessary to interpret the transmitted information and therefore improve predictions (Burgos and Polani, 2014, 2015). Here, we keep this assumption with respect to the agents, and we study the effects of introducing a parasite in a population that previously evolved its codes as well as its structure.

Model

In our previous model of code evolution (Burgos and Polani, 2014, 2015), the outputs or messages of an agent were produced according to a code, which was a conditional probability from sensor states to messages. The probability of each sensor state of an agent conditioned on the environmental variable μ was given. The information about the environment of each agent was obtained by considering the mutual information between the environmental variable and its sensor variable, together with the outputs of other agents. These outputs would be perceived or not, according to the structure of the population. The codes, as well as the population structure, were optimised in order to maximise what was called the similarity of the codes (we will introduce a more suitable term below) among the interacting agents of the population.

Here, instead, we consider a simplified model where the sensor states of an agent are the agent's messages, which are represented by a random variable X_Θ. That is, $p\left(X_\Theta \mid \mu, \Theta = \theta\right)$ gives the probability distribution of the sensor states (and, simultaneously, the messages) of an agent θ given the environmental conditions μ.

Agents perceive the sensor states (messages) of other agents according to the structure of the population, which is given by $p(\Theta, \Theta')$. This joint probability induces a weighted graph, where agents represent the nodes of the graph and there is an edge from agent θ to an agent θ' if $p(\theta, \theta') > 0$ (which is the weight of the edge). We interpret $p(\theta, \theta')$ as the probability of interaction between these two agents, and thus we require that $p(\theta, \theta') = p(\theta', \theta)$ (interactions are symmetrical) and $p(\theta, \theta) = 0$ for every agent θ (self-interaction is excluded).

We now consider a population of agents where interacting agents maximise their *mutual understanding*. This is formally defined by $I\left(X_\Theta\,;\,X_{\Theta'}\right)$, and, when this value achieves its maximum, the mapping that results from the agents' codes, $p\left(X_\Theta \mid X_{\Theta'}\right)$, is deterministic. It is important to note here that this model allows the agents to cluster into different sub-populations due to differences in their codes. Therefore, each sub-population could have its own convention for representing different aspects about the environment, and the conventions used can be as varied as possible, as long as the mapping $p\left(X_\Theta \mid X_{\Theta'}\right)$ is universal among all sub-populations.

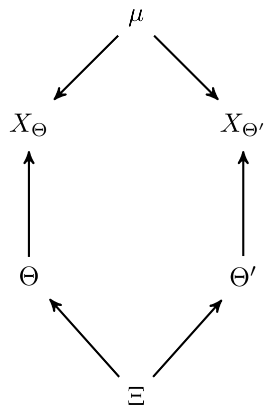


Figure 1: Bayesian network representing the relation of the variables in the simplified model of code evolution. $X_{\Theta'}$ is an *i.i.d* copy of X_Θ. Θ and Θ' selects agents from the same set, but their probability distributions are not necessary the same. These two variables depend on a common variable Ξ to model more general interaction structures.

For cases where the mutual understanding is locally optimal, the codes of the agents are related to each other in one of two possible manners: (a) within each sub-population, all agents have the same code, and the mapping $p\left(X_\Theta \mid X_{\Theta'}\right)$ is the identity matrix; or (b) within each sub-population, there are two types of agents (where the type is given by the agent's code), and the interaction is only between agents of different type. In this case, the graph induced by the interaction probability is bipartite between the types.

For cases where an agent interacts with more than one type of agent, then $p\left(X_\Theta \mid X_{\Theta'}\right)$ will necessarily be probabilistic, and thus the mutual understanding among the population will decrease. We can measure the total amount of information about the environment of an agent θ by $I\left(\mu\,;\,X_\Theta, X_{\Theta'} \mid \Theta = \theta\right)$. And, since the interaction probability is symmetric, the proposed measure for agent θ is equal to $I\left(\mu\,;\,X_\Theta, X_{\Theta'} \mid \Theta' = \theta\right)$. Let us note that, whenever the mutual understanding of a population is optimal, then the individuals that interact necessarily capture the same aspects of the environment. Then, at the optimum of mutual understanding in a population, agents do not increase their information by reading other agent's messages, although this indeed plays an important role in the evolution of codes. Nevertheless, the ties that an agent establishes are relevant for other purposes, which we study in the following sections.

Informational parasitism

There are different ways to define an informational parasite. Here, we adopt the model that characterises an informational parasite as an agent π that tries to minimise the mutual understanding between the agents with whom it interacts. An informationally antagonistic parasite is not typical for biology, as the parasite is concentrating at abusing the host system for its own interest, but does not care about the host except for avoiding detection. However, in the context of social networks or news sources, such a parasite can be considered a "troll" or a "disinformation" (FUD) agent who has direct interest in damaging the mutual understanding of the other agents of the population and/or their confidence in their knowledge of the true state of the environment.

In our case, the parasite will choose its code in such a way that the value $I(X_\Theta\ ;\ X_{\Theta'})$ is minimised. This is an extreme case of parasitism, where the parasite may kill its host as a result of maximising damage. Usually, the known parasites manipulate their hosts in order to benefit from it, decreasing their fitness such that it would not kill them (Schmid-Hempel, 1998). Although the defined type of parasite is not common in biology, it is still a possibility in the range of behaviours that decrease fitness of the host while increasing the attacker's fitness.

We now analyse the consequences of introducing a parasite in a population for a few very simple, but illustrative, scenarios. First, let us define the following types of codes:

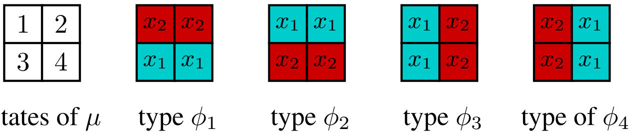


Figure 2: The left-most grid shows an illustration of the environment, although it does not denote its real structure. Then, each type shows a partition of the environmental states induced by an agent's code (codes here are deterministic). The types ϕ_1 and ϕ_2 capture the first bit of μ. Types ϕ_3 and ϕ_4 capture the second bit of μ.

Let us analyse a few simple scenarios where a parasite attacks a population. Let us assume that two (non-parasitic) agents share the same code, for instance, agents θ_1 and θ_2 are of type ϕ_1, and that these agents interact only with each other (their mutual understanding is 1 bit). Now, if we minimise their mutual understanding by introducing a parasite π, the optimal configuration is the following: the parasite interacts with one agent only and the parasite's code is of type ϕ_2 (the "opposite" of type ϕ_1). The mutual understanding between all three agents now is zero. Let us note that, in this case, the environmental information of each agent, before and after introducing the parasite, is 1 bit, which is the amount of information each of them acquire through their corresponding sensors.

Let us consider now a more interesting case: two sub-populations of two agents each, where agents only interact with agents within the sub-population and where the codes are the following: in the first sub-population, we have agents θ_1 and θ_2 of code type ϕ_1 (they capture the first bit of μ); and in the second sub-population, we have agents θ_3 and θ_4 of code type ϕ_3 (they capture the second bit of μ). Now, when we introduce a parasite, two configurations achieve zero mutual understanding: in both, the parasite interacts with every agent, but in the first one, it adopts the "opposite" code of agents θ_1 and θ_2, which is code type ϕ_2, while in the second one, it adopts the "opposite" code of agents θ_3 and θ_4, which is code type ϕ_4. Here, in the first case, the environmental information of agents θ_3 and θ_4 increases, since the parasite conveys information captured by agents θ_1 and θ_2 (captured by the parasite through its "opposite" code) that the former two agents do not possess. In the same way, for the second configuration, agents θ_1 and θ_2 are benefited by their interactions with the parasite, since here also, the parasite conveys information they lack.

Methods

All optimisations were performed using a genetic algorithm (GA). We utilised the C++ library GAlib v2.4.7 (Wall, 1996). The GA searches for a particular (possibly local) optimum, and this optimum corresponds to an evolutionary process that has converged. In order to accelerate computations, we assume the following: the probability distribution over μ is uniform, the codes of all agents are deterministic, and the interaction probability between any two agents is given by one over the amount of interactions. To visualise the evolution of the codes of the agents, we use the method of multidimensional scaling provided by R version 2.14.1 (2011-12-22). This method takes as input the distance matrix between codes, and plots them in a two-dimensional space preserving the distances as well as possible.

Results

We study the introduction of a parasite in a population where the mutual understanding was previously maximised. We consider a population of 256 agents, in an environment with

16 equally likely states, where agents can encode the environment using 16 different symbols. In this way, agents can potentially capture by themselves all the information about μ. As a result of the optimisation process, we obtained 5 sub-populations, where, in each of them, the induced interaction graph is bipartite. Therefore, there are 10 different codes globally, two per sub-population. In Fig. 3 we show the distance between the resulting codes, with point's size proportional to the number of agents that adopted each code. The distance used is the Jensen-Shannon divergence (see (Burgos and Polani, 2014)). The average mutual understanding in the population is $I\left(X_\Theta\ ;\ X_{\Theta'}\right) = 3.93$ bits, which, coincidentally, is also the mutual understanding within each sub-population (this does not need to hold necessarily).

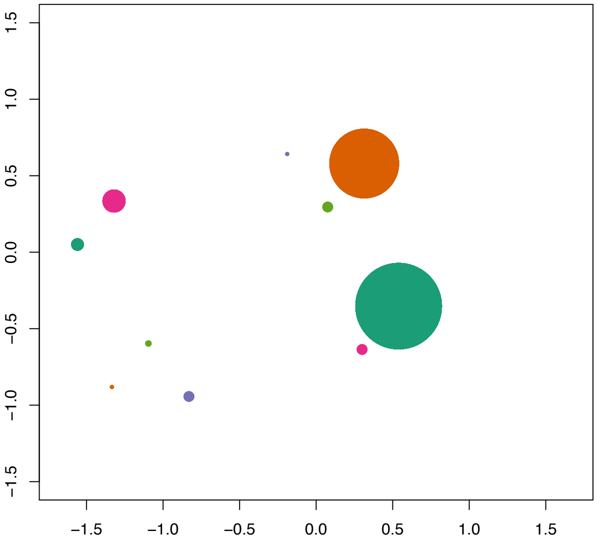
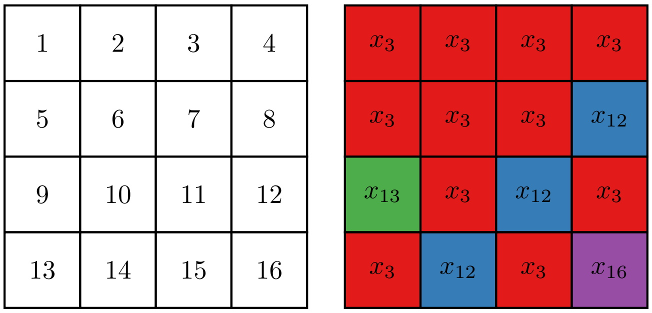


Figure 3: 2-dimensional plot of the distance between the codes: each point represents a particular code, and its size is relative to the number of agents adopting that particular code. The colour of the points denotes the sub-population to which the codes belong.

Blending in with the crowd

We introduce now a parasite π in the population, and we let it freely choose with whom it interacts, as well as its code (the parasite is introduced before the optimisation process begins, at generation 0). However, the parasite is allowed to use 32 symbols to encode μ, instead of 16, as we did for the rest of the agents. The reason for allowing a larger set of symbols to the parasite is that, otherwise, we will be forcing the parasite to use the symbols used by the population. We allow the double amount of symbols to enable the parasite to perfectly encode the environment while avoiding all symbols already in use.

We found that after optimisation, the parasite interacts with every agent of the population, and its code distance to every other agent is maximal (the distance is 4, the maximum achievable with 16 states). The resulting average mutual understanding now is $I\left(X_\Theta\ ;\ X_{\Theta'}\right) = 2.55$ bits (before the attack, it was 3.93 bits), and the code of the parasite is shown in Fig. 4.



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

(a) states of μ

x_3	x_3	x_3	x_3
x_3	x_3	x_3	x_{12}
x_{13}	x_3	x_{12}	x_3
x_3	x_{12}	x_3	x_{16}

(b) code of π

Figure 4: (a) Illustration of the environment μ, although the grid does not denote its real structure. (b) Partition of the environmental states induced by the code of the parasite π.

To understand the choice of code by the parasite, we analyse the joint probability $p(X_\Theta, X_{\Theta'})$ before introducing the parasite in the population. Our results show that the parasite encodes the environment through the messages that are most commonly used among the population. In this case, the parasite chose 4 messages (x_3, x_{12}, x_{13} and x_{16}), all of them among the most popular in the population (see Fig. 5).

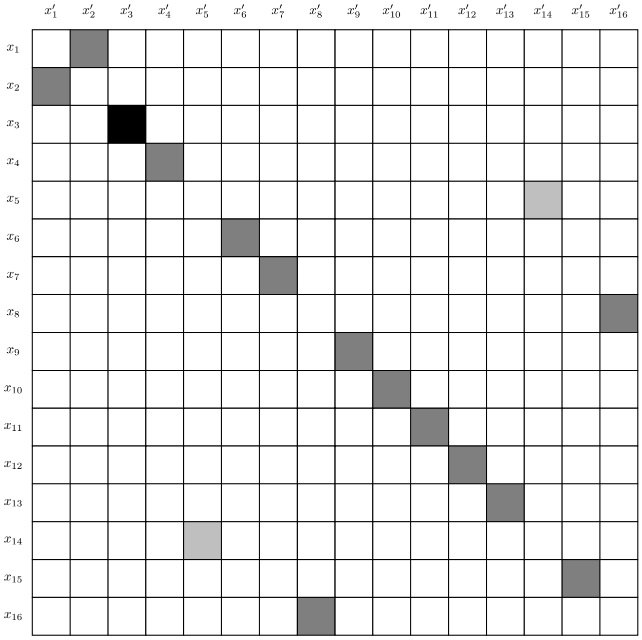


Figure 5: Joint probability of messages $p(X_\Theta, X_{\Theta'})$ of the population before the parasitic attack. Values are normalised to the maximum of all values, in this case $p(x_3, x'_3) = 0.125$. White squares have probability zero.

As a consequence of this antagonistic behaviour, the parasite blends in the population. This suggests that the parasite would try to avoid being identified by its messages. Our model allows us to measure how "identifiable" agents are by comparing the average joint messages. For instance, this can be measured by the mutual information between the agent selector and the joint messages, $I(\Theta \mid X_\Theta, X_{\Theta'})$. For a population with a universal code, this measure is zero, that is, we cannot identify any agent. Here, we want to know particularly how much we can identify the parasite by its messages. Then, we can consider the following measure:

$$D_{KL}\left(p(X_\Theta, X_{\Theta'} \mid \Theta = \pi) \,||\, p(X_\Theta, X_{\Theta'})\right) \qquad (1)$$

In Fig. 6, we show the values of Eq. 1 during the optimisation process, which shows that the divergence diminishes as the parasite minimises the mutual understanding of the population.

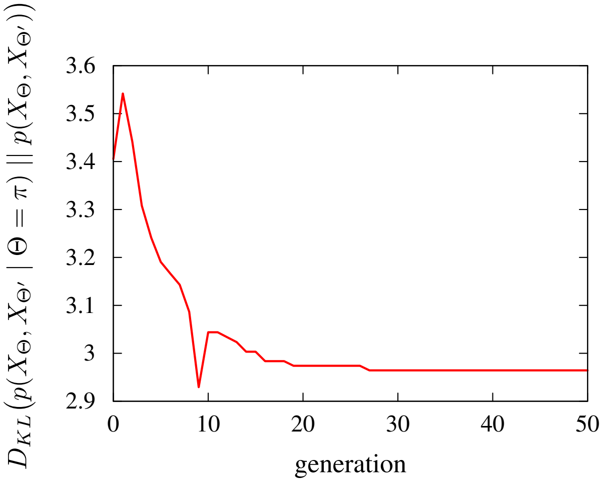


Figure 6: We measure how much we can identify the parasite from its messages. As a consequence of maximising damage to the population, the parasite blends in. The parasite is introduced at generation 0.

Missing environmental aspects

However, it is not only a question of choosing common messages: they must not coincide with other agent's codes given the environmental conditions. In other words, these messages that are popular among the population will be used by the parasite to express different aspects of μ. Otherwise, if the parasite expresses overlapping aspects, then there might be coincidences with one or more sub-population's adopted conventions. Consequently, the parasite will capture missing aspects in the population. This can be measured by how much information the code of the parasite adds about μ if we look at the average messages. Formally,

$$I\left(\mu\, ;\, X_{\Theta'} \mid X_\Theta, \Theta' = \pi\right) \qquad (2)$$

The value of Eq. 2 is plotted in Fig. 7 during the optimisation process. After convergence, we have that $I(\mu\, ;\, X_{\Theta'} \mid X_\Theta, \Theta' = \pi) = 1.30$ bits, while what the parasite acquires, through its sensors only, is $I(\mu\, ;\, X_\Theta \mid \Theta = \pi) = 1.32$ bits. That is, almost all the information it captures is missing in the population. If we consider the perceived information from μ together with the environmental information provided by the population, the parasite captures $I(\mu\, ;\, X_\Theta, X_{\Theta'} \mid \Theta = \pi) = 4$ bits, which is the maximum possible, and this means that the parasite always correctly predicts the environment.

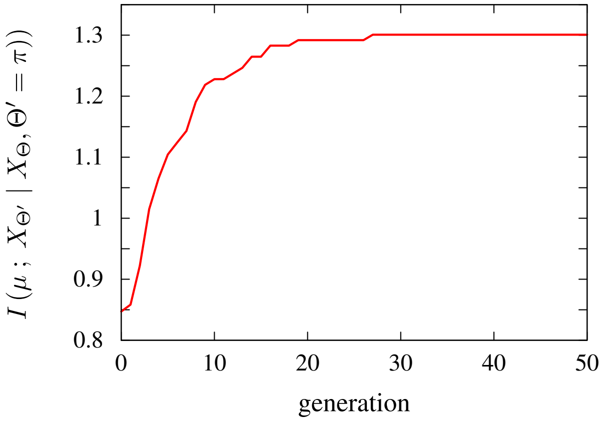


Figure 7: Amount of information the parasite possesses that the population lacks. The parasite is introduced at generation 0.

Robustness against parasites

After the parasitic attack, each sub-population has diminished its mutual understanding by different quantities. Although the (former) sub-populations now share a common agent (the parasite, and thus are not strictly speaking different sub-populations), we maintain the colours used in Fig. 3 to identify them. In Table 8 we show a summary of the outcome of the parasitic attack.

As Table 8 shows, in general, larger sub-populations are less damaged by a parasitic intrusion. This phenomenon is due to the large number of interactions among friendly agents, which diminishes the influence of any single agent by considering the average of the perceived messages. The exception in the example is the second largest sub-population, which becomes more damaged than the third largest sub-population. The reason why we see this is that the former

Colour	Size	Sizes of types	I_1	I_2	I_3
🟩	47	$41, 6$	3.93	2.85	3.59
🟧	35	$33, 2$	3.93	2.15	3.05
🟪	16	$11, 5$	3.93	2.54	3.33
🟥	8	$5, 3$	3.93	2.16	2.98
🟩	7	$5, 2$	3.93	1.83	2.65
	113	$95, 18$	3.93	2.55	3.50

Figure 8: Summary of the parasitic attack for each sub-population. The colours of each sub-population are the same as the ones in Fig. 3. I_1 is the mutual understanding before the parasitic attack, I_2 is the mutual understanding after the parasitic attack, and I_3 is the mutual understanding after the population's response.

sub-population is highly unbalanced, having a small number of agents of one type. Then, agents of the most numerous type interact only with a small number of agents, and therefore are more vulnerable to malicious agents.

Another way in which a population can defend itself against parasitic attacks is through diversification of their codes. Particularly, agents can reduce damage by using synonyms to express the same conditions. The presence of synonyms presents an obstacle for the parasite: when trying to confuse agents by expressing different conditions with a chosen symbol, the meaning of the correspondent synonym is not obfuscated.

We study this by comparing populations with different amounts of code types, while maintaining the same population structure. The setup is the following: the population is well-mixed (every agent interacts with every other agent), and first we randomly sample a code for every one of the 64 agents with symbols in the range $[1, 16]$. This population has one type of code only, and the used sample has a mutual understanding of 3.5 bits. Then, we produce a new population by modifying the code of half of the agents, such that $p\left(x + 16 \mid \mu\right) := p\left(x \mid \mu\right)$ and then we set $p\left(x \mid \mu\right) := 0$ (here, $x + 16$ is a synonym of x). In this way, the mutual understanding remains the same for the modified population, which now has two types. Each new type is introduced in a similar fashion, always mapping the original code to a set of (16) unused symbols.

We perform the minimisation of the mutual understanding on each population by introducing a parasite until convergence. We show in Fig. 9 the values of $I(X_\Theta \ ; \ X_{\Theta'})$ during the optimisation process for each population. Our results confirm our expectations: more diverse populations are more resistant to parasitic attacks.

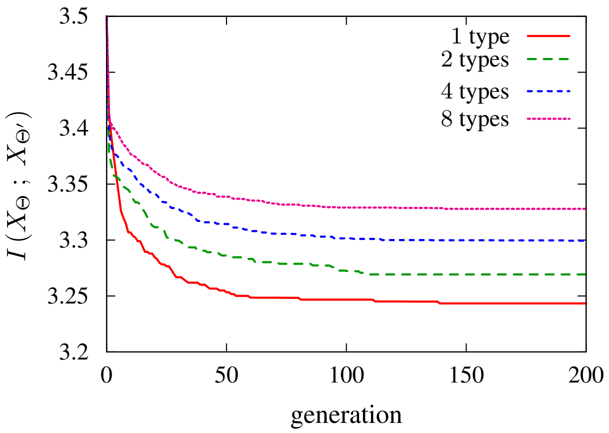


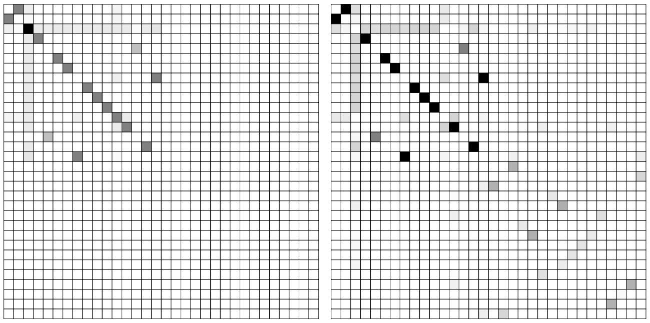
Figure 9: Mutual understanding of populations with varying amounts of types of codes during optimisation. Populations with more types of codes are more resistant to parasitic attacks. In all cases, the parasite was introduced at generation 0.

Code diversification

Now we let the population respond to the parasitic attack. If we let the rest of the agents respond to the parasite by freely changing the structure of the population, then our simulations shows that the parasite becomes isolated from the population, which is the expected outcome. However, we consider here a scenario where the structure of the population is maintained, and agents can only respond to the attack by updating their codes. In the same way as we did with the parasite, we allow the agents to choose from a larger set of messages (we consider 32 symbols to give the option to agents of changing completely their encoding of μ).

After convergence, the population's mutual understanding recovered to a value of $I\left(X_\Theta \ ; \ X_{\Theta'}\right) = 3.50$ bits (see Table 8 for a summary). In response to the parasitic attack, the agents of the population replaced, mostly, the symbols utilised by the parasite with unused ones. In Fig. 10, we can see how the joint probability $p(X_\Theta, X_{\Theta'})$ changed after the population's response. Here, the symbols present in the parasite's code (x_3, x_{12}, x_{13} and x_{16}) are mostly removed from the population's codes.

Three important features follow from the population's response: first, new code profiles are created in the population. For instance, the orange, purple and green sub-populations shown in Fig. 3 now consist of three types of codes (see Fig. 11. Nevertheless, the bipartite property is kept, but, instead, synonyms were adopted by one type in these sub-populations. This is due to the large amount of symbols available that, in the case they are not in use within the agent's type, are detached from any meaning and thus would not create confusion.



(a) after the parasitic attack.

(b) after the population's response.

Figure 10: Joint probability of messages $p(X_\Theta, X_{\Theta'})$ (a) after the parasitic attack and (b) after the population's response.

This can be appreciated in Fig. 12, where we represent the code of an agent before and after the population's response to the parasitic attack. This agent updated its code such that most symbols used by the parasite are avoided (x_3 and x_{12} are changed for x_{29} and x_{21}, respectively). On the other hand, x_{13} is kept. To check whether this is an optimal solution, we manually updated the code of all agents of the same type, changing x_{13} with every other possible symbol. Indeed, using this particular symbol occupied by the parasite maximises the population's mutual understanding. The reason for this is that, since all other symbols are occupied by more than one agent, x_{13} is the one that confuses the population the least.

Second, by drifting from the parasite's symbols, agents may update their codes in such a way that, after the update, they capture more environmental information. This is the case of the type shown in Fig. 12: before the update, environmental states 9 and 16 (see Fig. 4a to locate these states) were represented by x_3, while after the update, these states are distinguished from one another.

Third, and most important, the information that the parasite offers can now be understood (although not entirely) by the population: the missing information is mostly expressed using symbols that are not occupied any more by the agents of the population. This cannot be shown in the example, since changes in the agent's codes after the response to the attack may result in an overlap with the information that the parasite captures. However, we can manipulate the resulting configuration after the parasitic attack to show that agents now consider the information offered by the parasite. For each agent that is not the parasite, we update its code such that $p(x + 16 \mid \mu) := p(x \mid \mu)$ and then we set $p(x \mid \mu) := 0$. In this way, we make sure that all agents capture the same aspects of μ as before the update, without interference (all of the parasite's symbols are in the range $[1, 16]$).

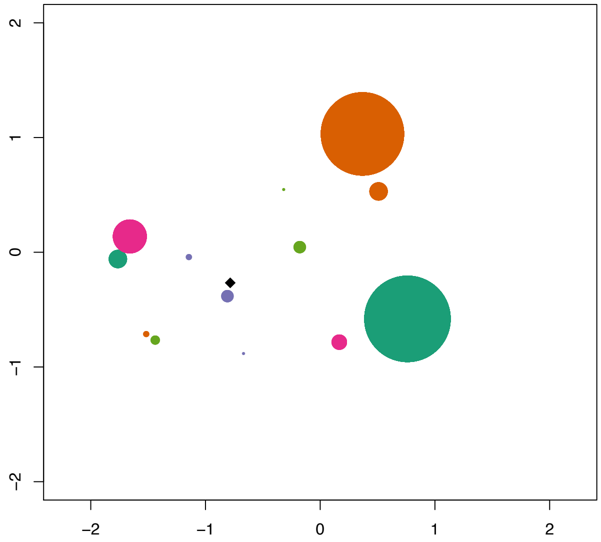


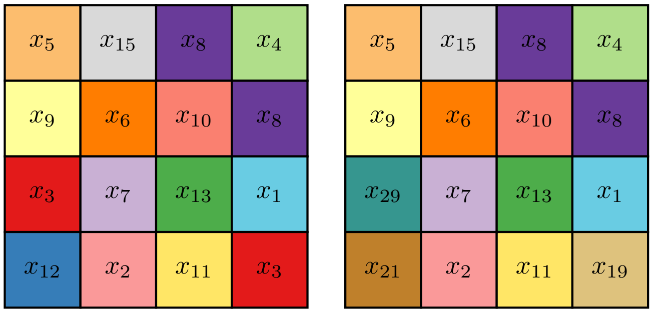
Figure 11: 2-dimensional plot of the distance between the codes after the population's response to the parasite attack. Each point represents a particular code, and its size is relative to the number of agents adopting that particular code. The colour of the points denotes the sub-population to which the codes belong. The black diamond represents the parasite's code.

Now, we measure the average environmental information before and after the change. Before, the value was $I(\mu\ ;\ X_\Theta, X_{\Theta'}) = 3.70$ bits, and after, $I(\mu\ ;\ X_\Theta, X_{\Theta'}) = 3.72$ bits. The increase is small, but we are considering one parasite only. If we introduce 8 parasites in the population, then the increase in the average environmental information is more significant: from 3.43 bits to 3.73 bits after updating the codes. It is worth noting that, if the parasites interact between themselves, then they would try to capture not only environmental information that is not present in the population, but also that is not captured by the other parasites.

Discussion and conclusions

We have considered a scenario where a parasite is introduced in a previously evolved population, and, after convergence, we looked at the response of the population, in one step of many in the co-evolutionary arms race. We considered one step only since, in this setting where the agent's behaviours are not unified, the arms race will cycle continuously.

Our model shows interesting behaviour consistent with empirical observations. For instance, parasites are known to mimic the chemical signatures utilised by the attacked host (D'Ettorre et al., 2002; Lorenzi et al., 2014). In this way, identification of the parasite by the population becomes harder. We measured this property during the parasite's attack, showing that as it increased damage in the mutual understanding of the population, it blended in. Additionally,


x_5 x_{15} x_8 x_4
x_9 x_6 x_{10} x_8
x_3 x_7 x_{13} x_1
x_{12} x_2 x_{11} x_3
x_5 x_{15} x_8 x_4
x_9 x_6 x_{10} x_8
x_{29} x_7 x_{13} x_1
x_{21} x_2 x_{11} x_{19}

(a) before the parasitic attack

(b) after the parasitic attack

Figure 12: Partition of the environmental states induced by the code of an agent (a) before, and (b) after the parasitic attack.

we showed that it becomes parasitically dependent on the population, as most of the environmental information it uses to predict the environment comes from the population.

We have also showed which properties a population may have in order to be robust against parasitic attacks. For instance, large populations are more resilient, since its numerous members provide a solid standard from which perturbations become less significant. Another way in which the population becomes resilient is for the population's agents to utilise synonyms. If the parasite intends to create confusion among the population by using messages that have a different meaning for the rest of the agents, then when synonyms are present, then they do not present any ambiguities.

The presence of parasites in a population can be, in the long term, a positive force (Hudson et al., 2006). For instance, they increase the diversity of the population, which in our scenario was manifested in the creation of new types of codes by using synonyms. As we have seen, this makes the population more robust to subsequent attacks in their co-evolution. Second, parasites are able to capture information about the environment which is not captured by any other agent. Most of this information is not understandable by the agents until they respond to the parasitic attack by drifting their codes.

The code drift has two effects: first, it makes the parasite easier to identify, since it is the only agent using a particular set of messages; and second, after the messages used by the parasite are avoided, the parasite's information becomes understandable for the whole population. Therefore, after the population recovers from the attack, agents can improve their predictions of the environment. The parasite, after the population's response, can still perfectly predict the state of the environment, but with one major drawback: it becomes easily identifiable, and thus the population have the possibility to take action (for instance, by avoiding interaction) when a future attack begins.

References

Ackley, D. H. and Littman, M. L. (1994). Altruism in the evolution of communication. In *Artificial life IV*, pages 40–48.

Barabási, A.-L., Freeh, V. W., Jeong, H., and Brockman, J. B. (2001). Parasitic computing. *Nature*, 412(6850):894–897.

Brockhurst, M. a., Rainey, P. B., and Buckling, A. (2004). The effect of spatial heterogeneity and parasites on the evolution of host diversity. *Proceedings. Biological sciences / The Royal Society*, 271(April 2003):107–111.

Burgos, A. C. and Polani, D. (2014). An informational study of the evolution of codes in different population structures. In *ALIFE 14: The Fourteenth Conference on the Synthesis and Simulation of Living Systems*, volume 14, pages 352–359.

Burgos, A. C. and Polani, D. (2015). An informational study of the evolution of codes and of emerging concepts in population of agents. *Accepted to "Artificial Life"*.

D'Ettorre, P., Mondy, N., Lenoir, A., and Errard, C. (2002). Blending in with the crowd: social parasites integrate into their host colonies using a flexible chemical signature. *Proceedings. Biological sciences / The Royal Society*, 269(1503):1911–8.

Donaldson-Matasci, M. C., Bergstrom, C. T., and Lachmann, M. (2010). The fitness value of information. *Oikos*, 119(2):219–230.

Doyle, J. C. (2010). The Architecture of Robust, Evolvable Networks. *The Lee Center*, pages 12–14.

Hudson, P. J., Dobson, A. P., and Lafferty, K. D. (2006). Is a healthy ecosystem one that is rich in parasites? *Trends in Ecology and Evolution*, 21(7):381–385.

Kelly, J. (1956). A new interpretation of information rate. *IEEE Transactions on Information Theory*, 2(3):185–189.

Lorenzi, M. C., Azzani, L., and Bagnères, A. G. (2014). Evolutionary consequences of deception: Complexity and informational content of colony signature are favored by social parasitism. *Current Zoology*, 60:137–148.

Robbins, P. (1994). The effect of parasitism on the evolution of a communication protocol an artificial life simulation. In *Proceedings of the third international conference on Simulation of adaptive behavior: from animals to animats 3: from animals to animats 3*, pages 431–437. MIT Press.

Schmid-Hempel, P. (1998). *Parasites in social insects*. Princeton University Press.

Seger, J. and Brockmann, H. J. (1987). What is bet-hedging? *Oxford surveys in evolutionary biology*.

Shannon, C. (1948). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Wall, M. (1996). Galib: A c++ library of genetic algorithm components. *Mechanical Engineering Department, Massachusetts Institute of Technology*, 87:54.

Woese, C. R. (2004). A new biology for a new century. *Microbiology and Molecular Biology Reviews*, 68(2):173–186.

Simulation-Based Analysis of *in Situ* Cellular Motility

Mark Read[1,*], Jon Timmis[2] and Tatyana Chtanova[3]

[1]Charles Perkins Centre, The University of Sydney, Australia. *mark.read@sydney.edu.au
[2]Department of Electronics, The University of York, UK. [3]The Garvan Institute, Sydney, Australia.

A fundamental challenge in simulation-based investigation of biological systems is deciding how detailed to make the simulation. Here we present a methodology that uses an automated simulation parameterisation technique to guide simulation development to an appropriate level of abstraction. We demonstrate the methodology in the context of simulating cellular motility. Recent advances in modern day imaging technology allow for the *in situ* examination of individual cells' migration through tissues. Complex and sometimes highly coordinated patterns have been observed, such as the swarming response of neutrophils to sterile tissue injury [1]. It is not clear what factors drive these responses, but the detailed single-cell spatial-temporal location data that these microscopy technologies provide permits a simulation-based analysis approach.

Our simulation-based methodology attempts to recreate the dynamics observed *in situ* by providing a platform wherein models of cellular behaviour can be created and calibrated, and evaluated against real-world data sets. Our methodology is depicted in figure 1. Through an agent-based simulation approach cells are explicitly represented as individuals with unique internal states and locations in a spatial environment. A model of the behavioural responses of simulated cells to the stimuli they observe in their local environments can be specified. The location of each cell in the environment over time can be tracked, permitting the construction of a statistical profile of their motility dynamics. Similar profiles are constructed for the *in situ* data, and their direct comparison allows for the evaluation of how well the behavioural responses programmed into simulated cells reflect the real-world dynamics.

Accurately capturing real-world dynamics requires both an appropriate model, and appropriate parameters for that model. We employ multi-objective optimisation to identify the parameter sets that, given a particular model of cellular behaviour, best align simulation dynamics with those observed *in situ*. The metrics comprising the statistical profile, which might include the speeds cells move at and the rotational velocities at which they change direction, form objectives for calibration. NSGA-II [2] is used as our multi-objective optimisation algorithm. Where NSGA-II is unable to find parameter values that align simulation dynamics with those observed *in situ*, one may conclude that the underlying model of cellular behaviour is incorrect.

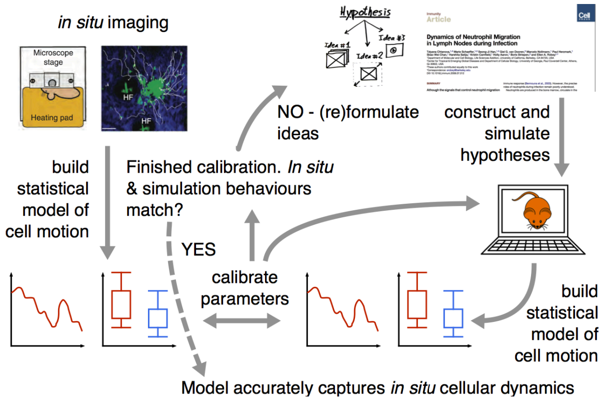

in situ imaging
Microscope stage
Heating pad
HF
HF
Hypothesis
Idea #1
Idea #2
Idea #3
Immunity
Article
Cell
Dynamics of Neutrophil Migration in Lymph Nodes during Infection
NO - (re)formulate ideas
construct and simulate hypotheses
build statistical model of cell motion
Finished calibration. *In situ* & simulation behaviours match?
YES
calibrate parameters
build statistical model of cell motion
Model accurately captures *in situ* cellular dynamics

Figure 1: Our simulation-based methodology for conceiving and evaluating hypotheses concerning the factors driving cell motility as observed through *in situ* imaging.

Thus far we have used this methodology to analyse non-directional neutrophil motility in the extra-vascular dermis of the mouse ear, in absence of tissue injury. We have concluded that neutrophils do not follow a Levy-flight motility pattern, wherein cells move in a uniformly random direction for a distance drawn from a long-tailed distribution. Rather, a model wherein changes in neutrophil movement direction are drawn from normal distributions with zero mean, and speeds are drawn from normal distributions unique to each cell accurately reflects the observed *in situ* dynamics.

Having established a proof of principle for our methodology, further work will examine the factors driving neutrophil swarming in response to sterile tissue injury. Moreover, the methodology has potential application in other domains, aiding in model selection and informing the level of detail required to accurately capture the biology.

References

[1] Chtanova, T., *et al.* (2008). Immunity 29:487-496.
[2] Deb, K., *et al.* (2002). IEEE Trans on Evo Comp, 6(2):182-197.

Simulating the Influence of Diet on the Intestinal Microbiome Composition

Mark Read, Andrew J. Holmes, Madison Hartill-Law, Samantha Solon-Biet,
David Raubenheimer and Stephen J. Simpson
Charles Perkins Centre, The University of Sydney, Australia.
mark.read@sydney.edu.au

Diet is a driving factor in the emergence of western lifestyle disease, and of the gut microbe community (microbiota) composition, which in turn has been linked to a host of diseases. There are many dimensions comprising ‘diet’: *macro-nutrient distribution*, the proportion of protein, carbohydrate and fat in food; *energy density*, the ratio of calories to food weight; *intake pattern*, the times at which we eat, and our incorporation of periods of fasting; and *macronutrient source*, such as carbohydrates in the form of sugar, complex carbohydrates or fibre. The rational design of diet interventions requires the integration of all these dimensions, which is challenging to do experimentally.

We are developing a simulation that integrates these diet dimensions to investigate how diet drives microbiota community composition. The simulation is agent-based, and explicitly represents individual bacteria cells and their location in the gut. We assume bacteria require access to carbon (C) and nitrogen (N) in a ratio of 5.2:1, and one of these is always a limiting factor on growth; we assume all other nutrients are freely available. Bacteria internalise C and N from their specific substrates in the local environment. Internalised stores of C and N decay, representing the ‘cost of living’, and their absolute quantity determines a bacterium’s probability of division or death. The gut is represented as a 1-dimensional space, with bacteria and digesta progressed through peristalsis events. The microbiota community composition is defined in terms of six ‘functional guilds’, based on bacteria substrates, which may be diet-derived macronutrients or mucin glycoprotein host-secretions. Long-chain and shorter-chain carbohydrates are represented, as is casein; these are components of real diet formulations input to the simulation. Protein contains both N and C, whereas carbohydrates provide only C. The mucin secretion rate has been estimated based on empirical data. Guild members compete for substrates, but do not otherwise interact.

A real-world data set of 30 diets, comprising 10 macronutrient distributions and 3 energy densities, administered to 250 cages of mice is used to parameterize the simulation [2]. The averaged food consumption of each cage is known, and used to plot that cage’s location on a macronutrient landscape. Interpolation between the 250 simulated cage datapoints reveals how different diets promote or disadvantage particular guilds. For example, figure 1 shows the relative abundance (% of total bacteria) of an example guild across the space of diets comprising different ratios of carbohydrate to protein. This particular guild ferments wheat starch and diet-derived protein. It is most competitive on diets comprising roughly equal quantities of each. Attributing real bacteria to guilds is problematic, as many are functionally diverse and satisfy the conditions of several guilds. Nonetheless, the caecal contents of real mice across this diet landscape have been sequenced, and the resultant landscapes of bacteria that can be assigned a guild, such as *Akkermansia muciniphila*, match those of their corresponding simulated guild.

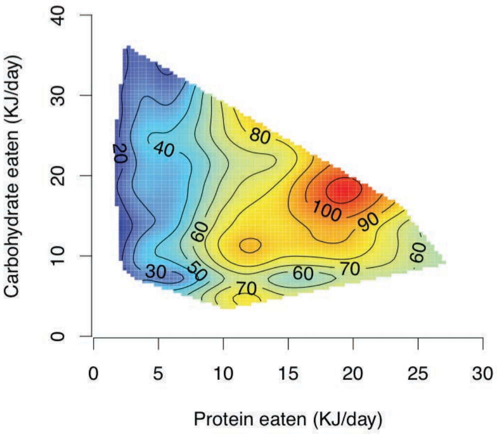


Figure 1: Relative abundance (%) of guild growing only on diet-derived protein and long-chain wheat starch carbohydrates across the diet landscape.

We have thus far examined how periodic fasting and the administration prebiotic fibres, both individually and in concert, shape the microbiota.

References

[1] Solon-Biet, S., *et al.* (2014). Cell Metabolism, 18:418-420.

Evolutionary change precedes extinction in eco-evolutionary dynamics based on a 3D virtual predator-prey system

Takashi Ito[1][2], Marcin L. Pilat[1], Reiji Suzuki[1] and Takaya Arita[1]

[1]Graduate School of Information Science, Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8601, Japan
[2]takashi@alife.cs.is.nagoya-u.ac.jp

Recent studies have reported that population dynamics and evolutionary dynamics, occurring at different time scales, can be affected by each other (Fussmann et al., 2007). Accepting not only that ecology affects evolution but also that evolution affects ecology lead to our recognition of the existence of eco-evolutionary (ecogenetic) feedbacks. We have been exploring the interaction between population and evolutionary dynamics using an Artificial Life approach based on a 3D physically simulated environment in the context of the predator-prey and morphology-behavior coevolution.

The first purpose of this study is to demonstrate coevolution between the predator and prey, and between morphology and behavior at the same time, by extending our previous model in which we evolved prey only to clarify the basic dynamics of eco-evolutionary feedbacks (Ito et al., ress). The second purpose is to explore the possibility of the extended model. We focus on the extinction, one of the most serious environmental issues, caused by the predator-prey and morphology-behavior coevolution. Specifically, we identify and elaborate a trait measure that can foresee an extinction of a species.

We use Morphid Academy as an open-source simulation system to evolve virtual creatures in a 3D physically simulated environment, in which the morphologies and behaviors of virtual creatures are evolved using a genetic algorithm. To represent the interaction between the group of predators and prey, we simulate every individual of both population pools in a shared environment (Ito et al., ress). Both species' traits and population size evolve concurrently. Each individual is evaluated on basis of 1) predation or escape ability and 2) cost of the body (it is incurred in proportion to an individual's body volume). They have chance to reproduce, with the expected number of offspring being determined by the sum of the evaluation value of the parents.

We observed the various patterns of extinction and looked for an index which changes when the ecosystem approaches extinction. We defined extinction as a sudden stop of the periodic increase and decrease in both population sizes (akin to the Lotka-Volterra system), followed by a state of very low (high) predator (prey) population size (near minimum (maximum) population size). We found a reasonable correlation between the remaining time until extinction and a simple index. The index we identified is the rate of change in the volume of body. For all cases of extinction observed in our experiments, first, we measured this index between two generations which are some generations apart from each other before extinction. Then we calculated the relation between this index of both species and the time from the generation to extinction. Analyzing 48 extinction events among 50 evolutionary experiments, we found that the remaining time until extinction was short when this index was large, and the time was long when this index was small. Specifically, calculating 95% confidence intervals, the shortest average remaining time until extinction was 151.7 ± 13.8 generations when the increase rate was $+0.15$ to $+0.20$, and the longest was 452.7 ± 12.3 generations when the rate was -0.60 to -0.65.

We hypothesize that the reason for this correlation is as follows: In our previous study, we found that the large body volume contributes to the good predation / escape strategies because the agent which has the large volume of their body can move faster and can guard from contact by predator to the specific body parts using other body parts. Both species acquire good strategies with high body volume cost as a result of the arms race between predation and escape abilities. However, a predator with a good strategy and a large body can offset the cost of its large body with its high predation success when the population density is high, but not when the population density is low. This causes an increase in the extinction risk of the predators when the population is in the low-density phase of its Lotka-Volterra dynamic. It shows that trait evolution can affect extinction risk in the predator-prey relationship in eco-evolutionary feedbacks.

We reported on the first step for searching the indices used for the prediction of extinction based on eco-evolutionary dynamics. The next focus must be whether or how broadly the simple index we found can be applied to the extinction events in nature.

References

Fussmann, G. F., Loreau, M., and Abrams, P. A. (2007). Eco-evolutionary dynamics of communities and ecosystems. *Functional Ecology*, 21(3):465–477.

Ito, T., Pilat, M. L., Suzuki, R., and Arita, T. (in press). Population and evolutionary dynamics based on predator-prey relationship in 3D physical simulation. *Artificial Life*, ALIFE 14 special issue.

Optimal Mutation Rate Control under Selection in Hamming Spaces

Elizabeth Aston[1], Alastair Channon[1], Roman V. Belavkin[2], Rok Krašovec[3] and Christopher G. Knight[3]

[1] School of Computing and Mathematics, Keele University, ST5 5BG, UK
[2] School of Engineering and Information Sciences, Middlesex University, London, NW4 4BT, UK
[3] Faculty of Life Sciences, The University of Manchester, M13 9PT, UK
{e.j.aston, a.d.channon}@keele.ac.uk, r.belavkin@mdx.ac.uk, {rok.krasovec, chris.knight}@manchester.ac.uk

Abstract

We investigate the effect of selection in a meta-genetic algorithm designed to optimize mutation rate control, based on the fitness of sequences relative to a defined optimum, in asexual evolution. Multiple innovations in the algorithm are required to achieve the evolution of optimal mutation rate control under selection. Before implementing selection, results from this improved algorithm clarify the optimal relationship of mutation rate to distance from the optimum as being a double sigmoid for binary sequences. Furthermore, the results clarify how such control functions depend on alphabet size, sequence length and the time horizon over which evolution is assessed. Incorporating selection leads to a distinctive shape of optimal mutation rate control function. This function has a mutation rate less that a third of 1/length at a Hamming distance of one from the optimum and beyond. This surprising result for a simple, universally monotonic single-peaked fitness landscape highlights the need for further research using models such as this. Future work will therefore explore how this control function may vary, for instance with population size and alternative selection mechanisms common in Artificial Life models.

Introduction

Evolution is dependent on the balance between mutation, recombination, selection and genetic drift. These processes can act in oposition to one another, for example mutation introduces variation while drift causes a reduction in diversity. In other circumstances they can act together, for instance mutation rate is positively correlated with recombination rate in human single nucleotide polymorphisms (Lercher and Hurst, 2002), in the Human Immunodeficiency Virus (Schlub et al., 2014), and in other retroviruses and RNA viruses (Tromas et al., 2014). Evolutionary processes can also interact in more complex ways, for instance, the type of recombination (crossover of single sequences versus crossover between diploid maternal and paternal sequences) can affect the magnitude of the critical mutation rate, above which sequences with greater robustness to mutation are favoured over individuals with greater fitness (Aston et al., 2013).

Genetic variation is at the centre of this balance. Both recombination and mutation introduce variation, while genetic drift and selection reduce it, effectively favouring certain individuals through random sampling or based on fitness or respectively. Selection and mutation in particular oppose one another in mutation-selection balance (Kimura and Maruyama, 1966; Bull et al., 2005). Selection affects the persistence of sequences depending on their fitness relative to other sequences in the population. The amount of variation maintained reduces as the strength of selection increases (Avila et al., 2014). Mutation rates and their levels relative to theoretically optimal and critical mutation rates, can determine how efficiently adaptation can occur and, to a degree, whether or not adaptations are lost.

These processes also interact with population parameters. For instance, genetic drift has a greater effect, relative to selection, in smaller populations due to random sampling. At a larger scale, Sung et al. (2012) show an inverse relationship between mutation rate and effective population size across the tree of life, including unicellular eukaryotes, eubacteria, and multicellular eukaryotes.

Together, these interactions occur in the context of a fitness landscape (Wright, 1932), in which each organism has a fitness value representing its relative replication rate (Domingo and Wain-Hobson, 2009). Under selection the population tends to move toward ‘fitter’ regions of the fitness landscape to form a quasispecies (Eigen and Schuster, 1979; Nowak, 1992; Bull et al., 2005; Nowak, 2006). In a landscape with a single fitness peak, a quasispecies is able to maintain its position surrounding the top of the peak, in mutation-selection balance, so long as the mutation rate does not exceed the error threshold above which adaptation is lost (Eigen et al., 1988; Nowak and Schuster, 1989; Tannenbaum and Shakhnovich, 2004; Bull et al., 2005; Saakian et al., 2006; Nowak, 2006; Takeuchi and Hogeweg, 2007; Domingo and Wain-Hobson, 2009; Schuster, 2009; Tejero et al., 2011). If the different evolutionary processes vary systematically across such a fitness landscape it has the potential to affect the course of evolution profoundly. Landscape-dependent control of evolutionary parameters has been considered in evolutionary algorithms (Eiben et al., 1999) but is much less understood in biology. For the key process of mutation, we have started to explore how its rate varies in practice (Krasovec et al., 2014). Here, for the first time, we consider how such rates might be controlled optimally, in simple fitness landscapes, in the presence of selection.

Mutation rate control in Hamming spaces

Mutation rate control has been studied theoretically and *in silico*. Belavkin et al. (2011) derived the probability of adaptation as a function of mutation rate, for populations of asexual self-replicating organisms with a monotonic fitness landscape, in the absence of selection. The probability of adaptation is dependent on the rate of mutation. This introduces the possibility that organisms may maximize the expected fitness of their offspring through mutation rate control. Belavkin et al. (2011) noted that, in general, analytical solutions are not available for long sequences evolving for more than one generation, and used a meta-genetic algorithm (Meta-GA) to evolve populations of mutation rate control functions. Each function $\mu(f)$ was used to control mutation rates in another GA, referred to as the Inner-GA. The Inner-GA constitutes the evolutionary model, while the Meta-GA acts to optimise mutation rate control.

The Inner-GA. The Inner-GA is a simple generational genetic algorithm in which each genotype is a sequence $\omega \in H^l_\alpha$, where $H^l_\alpha := \{1,\dots,\alpha\}^l$ is the space of sequences of length l and α letters and is equipped with the Hamming metric $d_H(a,b) := |\{i : a_i \neq b_i\}|$. Inner-GA populations, of 100 individuals each, are initialised by the Meta-GA (see below) and evolved by the Inner-GA with the objective being to maximise a fixed fitness function $f(\omega)$. Mutation is according to a per-locus mutation rate control function $\mu(f)$ specified by an individual genotype from the Meta-GA (below). In previous work (Belavkin et al., 2011) the Inner-GA ran for 500 generations and used no recombination and no selection, allowing for very rapid execution on GPGPU architectures due to thread independence.

The Meta-GA. The Meta-GA is a simple generational genetic algorithm that uses tournament selection (a good choice when little is known or assumed about the structure of the landscape) to maximise mean fitness in the final generation of the Inner-GA. Each genotype in the Meta-GA is a mutation rate control function $\mu(f)$, which is a sequence of real values $\mu \in [0,1]$ representing probabilities of mutation at different fitness values or binned intervals. In the case of Inner-GA fitness being the Hamming distance between sequences in H^l_α, Meta-GA genotypes have length $l+1$ so as to cover the range of fitnesses from 0 to l. Each run has a population of 100 functions, with each function initialised to $\mu(f) = 0$ for all f. In each generation of the Meta-GA, the Inner-GA is run once for each mutation rate control function (so 100 times), with each Inner-GA run evolving a population of 100 sequences.

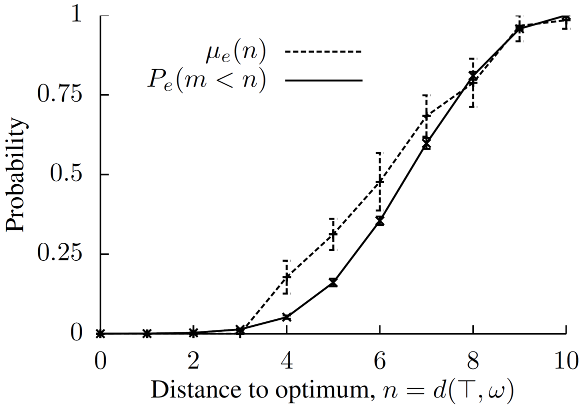


Figure 1: Mutation rate control function μ_e, evolved to maximise mean fitness $f(\omega) = -d_H(\top,\omega)$ after 500 generations of evolution without selection, $\omega \in H^{10}_4$. Averaged over 20 runs. Error bars show standard deviations. Taken from previous work (Belavkin et al., 2011, figure 3). The CDF P_e is also shown but not discussed in this paper.

The Meta-GA randomly selects three individuals from the population and replaces the least fit with a mutated single-point crossover of the other two. This process repeats until every individual in the Meta-GA population has been selected, or until fewer than three individuals remain, so producing the next generation. This method allows for excellent parallelisation, as triples can be selected at the start of each generation. The Meta-GA returns the fittest mutation rate control function $\mu(f)$ in its final generation.

In previous work (Belavkin et al., 2011), the Meta-GA ran for 500000 generations and its mutation procedure added a uniform-random number $\Delta\mu \in [-.1,.1]$ to one randomly selected value μ (mutation rate, bounded to be within $[0,1]$) in the individual mutation rate control function. In each generation of the Meta-GA, the 100 Inner-GA runs were seeded with identical populations. These were generated randomly, in each Meta-GA generation, so as to have the same number of individuals at each possible fitness value. Figure 1 shows the optimal mutation rate function previously found for short (length 10) base 4 sequences evolving for 500 generations without selection.

Mutation rate control in biological landscapes

The mutation rate control function evolved by the Meta-GA is dependent on the fitness landscape used in the Inner-GA. If fitness $f(\omega)$ corresponds to negative Hamming distance to the optimum, $-d_H(\top,\omega)$, then the optimal mutation rate can be seen to increase with $d_H(\top,\omega)$ (Belavkin et al., 2011). Biologically relevant landscapes are likely to be more complex than this simple case. In more rugged landscapes, fitness does not provide all necessary information about the position of a sequence in the landscape.

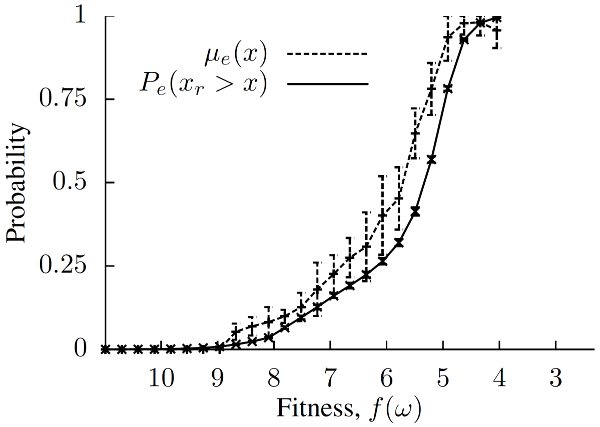


Figure 2: Mutation rate control function μ_e, evolved to maximise mean fitness on an aptamer landscape after 500 generations of evolution without selection, $\omega \in H_4^{10}$. Averaged over 20 runs. Error bars show standard deviations. Taken from previous work (Belavkin et al., 2011, figure 4).

There is extensive variation in mutation rates among species in nature (Sung et al., 2012). The degree to which biological organisms can control their mutation rate is a more open question, but variation in mutation rates within species and in response to the environment (mutation rate plasticity) could be widespread (MacLean et al., 2013; Krasovec et al., 2014). Such propensity for mutation rate variation in nature raises the question of whether the findings about optimal control of mutation rate in simple landscapes in silico could be extended to biological systems. Belavkin et al. (2011) evolved mutation rate control functions with fitness defined by an aptamer landscape. An aptamer is a nucleic acid that can be selected to bind to a particular target molecule (Knight et al., 2009). This complete DNA-protein affinity landscape was described by Rowe et al. (2010), and represents a rugged landscape with many local optima. Figure 2 shows the average of evolved mutation control functions for the aptamer landscape. This provides evidence that the simpler results for $f(\omega) = -d_H(\top, \omega)$ have relevance to biology. However, neither figure 1 nor figure 2 was produced using a model that included selection in the Inner-GA. Selection is a key aspect of evolution and introducing it into the model is therefore the next logical step.

Innovations in the Meta-GA

In order to achieve the results presented in this paper, it was necessary to make significant improvements to the Meta-GA. Most notably, the introduction of selection into the Inner-GA results in very much more rapid evolution to the target sequence, and so in Inner-GA populations spending very little search time far from the optimum. In the selection-based runs reported below, less than 1% of Inner-GA evaluations were at fitnesses in the lower quartile, and less than 5% were in the lower half. This necessary paucity of evaluations far from target greatly hinders the optimisation of mutation rates in these fitness regions. This is compounded by the requirement for inter-process communication and synchronisation, brought about by selection's need for fitness comparisons and the copying of genotypes between threads. This significantly reduces performance on GPGPU architectures, although these still provide a pronounced increase in performance when compared to a straightforward CPU implementation. In each run reported below, the Meta-GA was restricted (for practical considerations) to run for just 2000 generations.

Having explored a number of strategies for improving the Meta-GA's ability to optimise mutation rate control functions, and after searching over a wide range of parameters, the following modifications to the Meta-GA (as described in our previous work and this paper's introduction) were employed for the experiments reported here.

1. To evaluate a mutation rate control function from the Meta-GA, the Inner-GA is run not once but $10(l + 1)$ times, i.e. 110 times for $l = 10$ and 310 times for $l = 30$. The fitness of a mutation rate control function is summed over these runs.

2. Rather than seeding Inner-GA populations such that each begins with the same number of individuals at each possible fitness value, the multiplicity of runs introduced above is utilised further in order to improve the optimisation of mutation rates across the full range of fitness values. 10 runs are initiated with all individuals at the target ($d_H(\top, \omega) = 0$), and 10 runs are initiated with their populations randomised such that each individual is at distance 1, 10 runs at distance 2, ... 10 runs at distance l. Within each generation of the Meta-GA, all mutation rate control functions are evaluated using copies of the same $10(l+1)$ seed populations.

3. Uniform-random[1] *delta* mutation is reduced to $\Delta\mu \in [-.01, .01]$. This allows for better fine-tuning of mutation rates.

4. 10% of mutations in the Meta-GA are *resets* to a uniform-random value $\mu \in [1, 1]$; the other 90% remain *delta* mutations ($\Delta\mu$, see above). This aids in the optimisation of mutation rates at low fitnesses.

These innovations alone lead to interesting new findings about optimal mutation rate control in the absence of selection, as they generate clearer relationships between fitness and optimal mutation rate. Figure 3 demonstrates very

[1] Parallel sets of runs, to those reported below, employed Gaussian mutation over a range of mutation rates, with and without modification 4 (above). The results were consistent with those below but uniform-random mutation resulted in lower errors (variances) within the sets of 20 runs each.

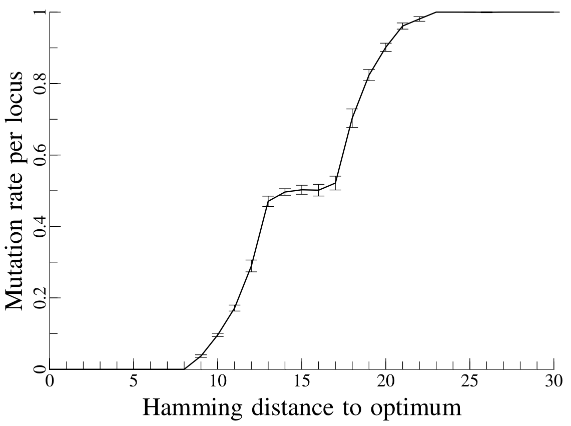


Figure 3: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 100 generations of evolution without selection, $\omega \in H_2^{30}$. Averaged over 20 runs. Error bars show standard deviations.

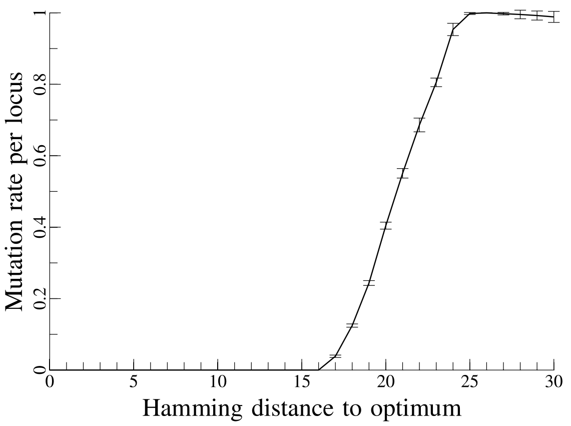


Figure 4: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 100 generations of evolution without selection, $\omega \in H_4^{30}$. Averaged over 20 runs. Error bars show standard deviations.

clearly that for a binary sequence, the optimal mutation rate control function is not a simple sigmoid but a double-sigmoid. Our intuition is that this may relate to binary's unique property, among alphabet sizes (α), that two mutations in succession, at the same locus, are guaranteed to return the original value. Certainly figure 4 shows that the optimal mutation rate control function for $\alpha = 4$ (as for DNA) is sigmoidal, not double-sigmoidal.

Figure 5, when compared to figure 4, shows that shortening sequence length from 30 to 10 results in a less sigmoidal, more "z-like" mutation rate control function; likewise comparing figures 7 and 6. Comparing figures 4 and 6 shows that reducing the number of (Inner-GA) generations before evaluation from 500 to 100 results in a steeper, more step-like optimal mutation rate control function, which rises above zero further from the optimum and more rapidly than for the less urgent function; likewise comparing figures 7 and 5.

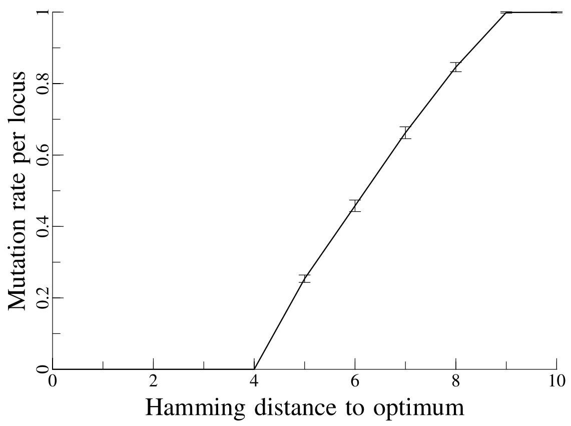


Figure 5: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 100 generations of evolution without selection, $\omega \in H_4^{10}$. Averaged over 20 runs. Error bars show standard deviations.

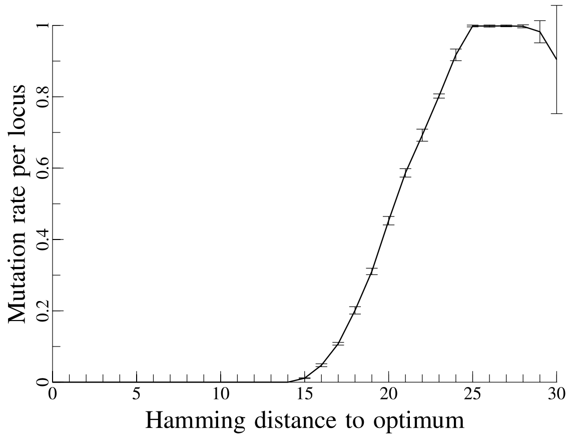


Figure 6: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 500 generations of evolution without selection, $\omega \in H_4^{30}$. Averaged over 20 runs. Error bars show standard deviations.

Note also that figure 7 is a match (with reduced error) for the evolved mutation function shown in figure 1, taken from (Belavkin et al., 2011), demonstrating that the innovations noted above have not affected the core result, other than reducing error.

Mutation rate control under selection

The main advantage of the innovations above are that they enable the evolution of mutation rate control functions un-

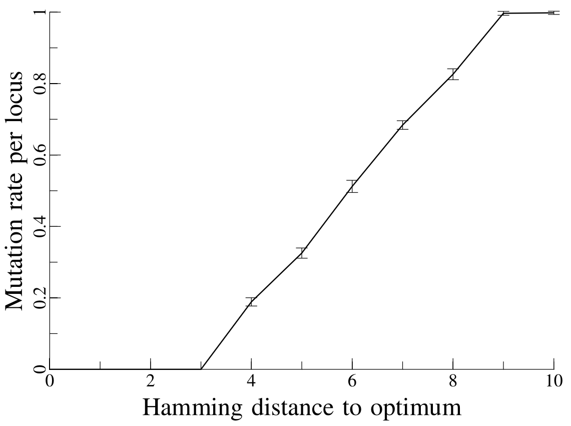


Figure 7: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 500 generations of evolution without selection, $\omega \in H_4^{10}$. Averaged over 20 runs. Error bars show standard deviations.

der selection, making this work much more relevant to both biology and general evolutionary algorithms. We introduce selection into the Inner-GA through the following modification: in every generation of the Inner-GA, each genotype is compared to a randomly chosen (per genotype) other genotype and replaced by a copy of that if that is fitter. Mutation is then applied, as for the without-selection case. The population is fully mixed.

Figure 8 shows the evolved mutation rate control function for 30-long sequences of base 4 (such as DNA) when evolving for 100 generations under selection. Values for highly unfit sequences (worse than random, i.e. $d_H > 22$) have not been highly optimised but the majority of the function shows very little variation among independent runs. There is a clear, distinctive shape to the optimal mutation rates moving out from the optimum. At the optimum the optimal mutation rate is of course zero. One mutation away it jumps to a non-zero value and then slowly rises from there. Most intriguingly, the optimal mutation rate one mutation from the origin is significantly lower than the "$1/L$" heuristic of conventional wisdom, despite the maximal simplicity of the fitness landscape.

The $1/L$ heuristic

It can be seen from figure 8 that the optimal mutation rate found for an Inner-GA that incorporates selection does not follow the $1/L$ heuristic. $1/L$ has been suggested as a general value for the per-locus mutation rate in a GA, where L is sequence length (Mühlenbein, 1992; Bäck and Schütz, 1996; Ochoa et al., 2000; Ochoa, 2002). Mühlenbein (1992) states that $\mu = 1/L$ is optimal for general unimodal functions. Ochoa (2002) used GAs with bit-strings to test the limitations of the $1/L$ heuristic. The change in optimal mutation rate with time was studied, along with the interaction between mutation rate, selection pressure, and population size. It was found that a mutation rate of $1/L$ produces optimal or near optimal results. It was also found that increasing the selection pressure increases the magnitude of optimal mutation rates; and that decreasing population size (without changing selection pressure) can result in a decrease in optimal mutation rate at small population sizes, specifically in the case of the Knapsack Problem when reducing population size from 50 to 10. It was concluded that a rate of $1/L$ is sub-optimal only when the selection pressure is either extremely weak or extremely strong, or when the population size is very small (Ochoa, 2002). The optimal mutation rate for Hamming distances close to the optimum in figure 8 is close to 1%, where as following the $1/L$ heuristic it is expected to be 1/30, i.e. approximately 3.3%. The fact that the observed optimal mutation rate is less than a third of the expected rate may relate to the small population size of 100 used in the Inner-GA, although population sizes of this magnitude are frequently used in Artificial Life work. The effect of tournament selection (also frequently used in Artificial Life work) on selective advantage could also be an explanatory factor.

Error threshold is related to optimal mutation rate, and is therefore also dependent on selection pressure. Ochoa et al. (2000) note that selection pressure is an informal term and that "Loosely the selection pressure measures the ratio of maximum to average fitness in the population". In a fitness landscape with a single peak with fitness σ >1, and with all other sequences having fitness 1, the error threshold, denoted by Ochoa et al. (2000) as p, is given as:

$$p = \frac{\ln(\sigma)}{L}$$

σ is the selective advantage of the master (optimum) sequence over all other sequences in the population, i.e. the selection pressure. L is the sequence length. In the simplest instance, σ represents the ratio of the reproduction rate of the master sequence to the average reproduction rate of the rest of the sequences in the population. Ochoa et al. (2000) looked at error thresholds and optimal mutation rates when solving both toy and real-world problems with a genetic algorithm. They concluded that error thresholds and optimal mutation rates are correlated. In addition, selection pressure was shown to have a significant effect on the magnitude of both the error threshold and the optimal mutation rate; the stronger the selection pressure, the greater the error threshold and optimal mutation rate (Ochoa et al., 2000). However, it has been observed that the error threshold has an exponential dependence on population size that is more noticeable in small populations (Nowak and Schuster, 1989; Ochoa and Harvey, 1998; Ochoa et al., 1999; Channon et al., 2011; Aston et al., 2013). The population in the Inner-GA may be considered small enough to be affected by the exponential

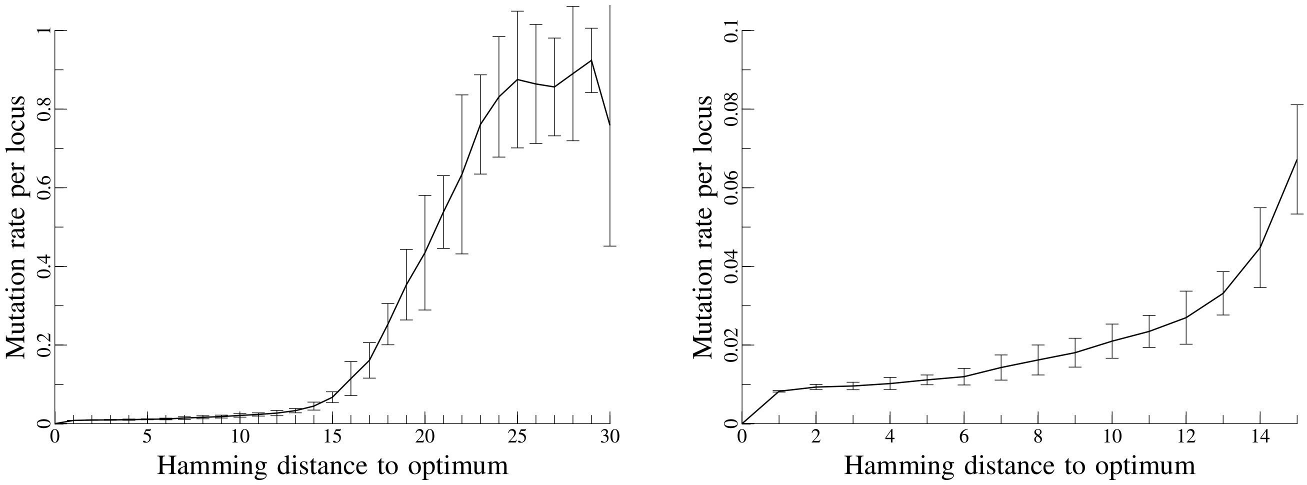


Figure 8: Mutation rate control function $\mu(f)$, evolved to maximise mean fitness $f(\omega) = -d_H(\top, \omega)$ after 100 generations of evolution under selection, $\omega \in H_4^{30}$. Averaged over 20 runs. Error bars show standard deviations. Shown in full (left) and in detail near the optimum (right).

model, potentially influencing both the error threshold and optimal mutation rate, and providing a possible explanation for the fact that the observed optimal mutation rate is less that a third of the expected rate following the erroneous $1/L$ heuristic.

Conclusions

The innovations in the Meta-GA have provided a demonstration that for binary strings the optimal mutation rate control function is a double sigmoid, and also enabled us to clarify the relationship of optimal mutation rate to both sequence length and number of generations in this scenario.

When applied to the evolution of mutation rate control under selection, we have successfully determined the distinctive shape of the optimal mutation rate control function. This has a mutation rate less that a third of 1/length at and near (above) a distance of one mutation from the optimum, despite the single-peak everywhere-monotonic simplicity of the fitness landscape. We have noted a number of possible reasons for this difference. This highlights a need for further research using models such as this, that can take population size and selection mechanisms common in Artificial Life models into account.

In previous work (Belavkin et al., 2011) we demonstrated that, in the absence of selection, the optimal mutation rate control function for a DNA-protein affinity fitness landscape resembles that for the monotonic case of fitness being negative Hamming distance. Further work is needed to evaluate whether or not the characteristics of optimal mutation rate control functions under selection generalise to such non-monotonic landscapes.

Future work may also incorporate more of the evolutionary pressures, namely recombination and genetic drift. It is expected that optimal mutation rate will counteract those pressures that reduce variation, and be complemented by those pressures that increase variation.

Acknowledgements

This work was supported by Engineering and Physical Sciences Research Council (EPSRC) grant EP/H031936/1 and Biotechnology and Biological Sciences Research Council (BBSRC) grant BB/L009579/1. The dataset underpinning the results is openly available from Zenodo at http://doi.org/4pv.

References

Aston, E., Channon, A., Day, C., and Knight, C. G. (2013). Critical mutation rate has an exponential dependence on population size in haploid and diploid populations. *PLoS ONE*, 8(12):e83438.

Avila, V., Pérez-Figueroa, A., Caballero, A., Hill, W. G., García-Dorado, A., and López-Fanju, C. (2014). The action of stabilizing selection, mutation, and drift on epistatic quantitative traits. *Evolution*, 68(7):1974–1987.

Bäck, T. and Schütz, M. (1996). Intelligent mutation rate control in canonical genetic algorithms. In *Proceedings of the Ninth International Symposiumon Foundations of Intelligent Systems*, volume 1079.

Belavkin, R. V., Channon, A., Aston, E., Aston, J., and Knight, C. G. (2011). Theory and practice of optimal mutation rate control in Hamming spaces of DNA sequences. In *Advances in Artificial Life, ECAL 2011:*

Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems.

Bull, J. J., Meyers, L. A., and Lachmann, M. (2005). Quasispecies made simple. *PLoS Computational Biology*, 1(6):e61.

Channon, A., Aston, E., Day, C., Belavkin, R. V., and Knight, C. G. (2011). Critical mutation rate has an exponential dependence on population size. In *Advances in Artificial Life, ECAL 2011: Proceedings of the Eleventh European Conference on the Synthesis and Simulation of Living Systems*.

Domingo, E. and Wain-Hobson, S. (2009). The 30th anniversary of quasispecies. *EMBO Reports*, 10:444–448.

Eiben, A. E., Hinterding, R., and Michalewicz, Z. (1999). Parameter control in evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 3(2):124–141.

Eigen, M., McCaskill, J., and Schuster, P. (1988). Molecular quasispecies. *Journal of Physical Chemistry*, 92:6881–6891.

Eigen, M. and Schuster, P. (1979). *The hypercycle*. Springer, New York.

Kimura, M. and Maruyama, T. (1966). The mutational load with epistatic gene interactions in fitness. *Genetics*, 54:1337–1351.

Knight, C., Platt, M., Rowe, W., Wedge, D., Khan, F., Day, P., McShea, A., Knowles, J., and Kell, D. (2009). Array-based evolution of DNA aptamers allows modelling of an explicit sequence-fitness landscape. *Nucleic Acids Research*, 37(1):e6.

Krasovec, R., Belavkin, R. V., Aston, J. A. D., Channon, A., Aston, E., Rash, B. M., Kadirvel, M., Forbes, S., and Knight, C. G. (2014). Mutation rate plasticity in rifampicin resistance depends on escherichia coli cell cell interactions. *Nature Communications*, 5:3742.

Lercher, M. J. and Hurst, L. D. (2002). Human SNP variability and mutation rate are higher in regions of high recombination. *Trends in Genetics*, 18(7):337–340.

MacLean, R. C., Torres-Barcelo, C., and Moxon, R. (2013). Evaluating evolutionary models of stress-induced mutagenesis in bacteria. *Nat Rev Genet*, 14(3):221–7.

Mühlenbein, H. (1992). How genetic algorithms really work: Mutation and hillclimbing. *Parallel Problem Solving from Nature*, 2:15–26.

Nowak, M. A. (1992). What is a quasispecies? *Trends in Ecology and Evolution*, 7:118–121.

Nowak, M. A. (2006). *Evolutionary dynamics: Exploring the equations of life*. Harvard University Press.

Nowak, M. A. and Schuster, P. (1989). Error thresholds of replication in finite populations: Mutation frequencies and the onset of Muller's ratchet. *Journal of Theoretical Biology*, 137:375–395.

Ochoa, G. (2002). Setting the mutation rate: Scope and limitations of the 1/l heuristic. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2002)*, pages 315–322.

Ochoa, G. and Harvey, I. (1998). Recombination and error thresholds in finite populations. In *Foundations of Genetic Algorithms (FOGA-5)*, pages 245–264.

Ochoa, G., Harvey, I., and Buxton, H. (1999). Error thresholds and their relation to optimal mutation rates. In *Proceedings of the Fifth European Conference on Articial Life*, pages 54–63.

Ochoa, G., Harvey, I., and Buxton, H. (2000). Optimal mutation rates and selection pressure in genetic algorithms. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2000)*.

Rowe, W., Platt, M., Wedge, D. C., Day, P. J., and Kell, D. B. (2010). Analysis of a complete DNA-protein affinity landscape. *Journal of Royal Society Interface*, 7(44):397–408.

Saakian, D. B., Muñoz, E., Hu, C., and Deem, M. W. (2006). Quasispecies theory for multiple-peak fitness landscapes. *Physical Review E*, 73:041913.

Schlub, T. E., Grimm, A. J., Smyth, R. P., Cromer, D., Chopra, A., Mallal, S., Venturi, V., Waugh, M. C., Mak, J., and Davenport, M. P. (2014). Fifteen to twenty percent of hiv substitution mutations are associated with recombination. *Journal of Virology*, 88(7):3837–3849.

Schuster, P. (2009). Genotypes and phenotypes in the evolution of molecules. *European Review*, 17(2):281–319.

Sung, W., Ackerman, M. S., Miller, S. F., Doak, T. G., and Lynch, M. (2012). Drift-barrier hypothesis and mutation-rate evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 109(45):18488–18492.

Takeuchi, N. and Hogeweg, P. (2007). Error-threshold exists in fitness landscapes with lethal mutants. *BMC Evolutionary Biology*, 7:15.

Tannenbaum, E. and Shakhnovich, E. I. (2004). Solution of the quasispecies model for an arbitrary gene network. *Physical Review E*, 70:021903.

Tejero, H., Marin, A., and Montero, F. (2011). The relationship between error catastrophe, survival of the flattest, and natural selection. *BMC Evolutionary Biology*, 11:2.

Tromas, N., Zwart, M. P., Poulain, M., and Elena, S. F. (2014). Estimation of the in vivo recombination rate for a plant RNA virus. *Journal of General Virology*, 95:724–732.

Wright, S. (1932). The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In *Proceedings of the Sixth International Congress on Genetics*, pages 355–366.

An Agent-Based Model of Avascular Tumor Growth

Ana Victoria Ponce Bobadilla[1], René Doursat[2] and François Amblard[3]

[1]Erasmus Mundus Master's in Complex Systems Science,
Graduate School, Ecole Polytechnique, Paris, France
[2]BioEmergences Lab, CNRS (USR3695), Gif-sur-Yvette, France
[3]Institut Curie, Paris, France
ana-victoria.ponce-bobadilla@polytechnique.edu

Abstract

We propose a simplified agent-based model of avascularized tumor. It involves a tissue in which blood vessels introduce nutrients that diffuse. Cells move, proliferate and die according to an individual quantity of "energy" and free space for their offspring. They can transition to a "cancerous" type and an intermediate "mutated" type, where they behave normally but can be affected by cancerous neighbors. We are interested in finding the key parameters that can lead a majority of cancerous cells to be replaced by normal ones. First, we give a brief overview of previous tumor growth models, especially in avascular tissues. Then, we describe in detail the agents and rules of our model, commenting on the choices made. Next, we conduct a parameter space exploration, varying in particular the influence of neighbors, the division probability and mutation probability. Results show a marked phase transition between domains of high cancerous cell density and high mutated cell density. We also analyze the importance of certain rules in our model by "rule knockout" and find that energy-dependency of division and space for offspring are essential, while type-specific division probabilities are not. Finally, we discuss the overall relevance of the model and possible future improvements.

Introduction

According to the World Health Organization, cancer is among the leading causes of death worldwide, with 8.2 million fatalities in 2012 (Stewart and Wild, 2014). Mathematical models are being created to help understand the underlying mechanisms of tumor growth, with the potential to create a framework for simulations and virtual experiments. This should enable scientists to observe the effects of different treatments more efficiently, and improve them or suggest new ones (Roose et al., 2007).

Cancer can generally be defined as the uncontrolled growth and spread of cells, also referred to as "malignant" tumors and "neoplasms". There are more than 100 types of cancers: lung, liver, stomach, colorectal and breast cancers are among the most lethal (Stewart and Wild, 2014). However, they all have certain characteristics in common: the tumor mass typically grows beyond its usual boundaries, invades neighboring parts of the body, then spreads out to remote organs.

It is still debated how exactly cancerous behavior is initiated. The general consensus is that several gene mutations are required to turn a normal cell into a cancer cell. The factors that trigger these mutations are largely unknown, but are thought to include both environmental and hereditary properties (Roose et al., 2007).

Once tumor cells have started to appear, tumor growth goes through three different stages:

- *Avascular growth:* This stage is characterized by the proliferation of cancer cells. The tumor forms a solid mass that expands by mitosis, depending largely on available nutrients. There is no invasion of healthy tissue yet. As nutrients deplete, the cancer cells die (necrosis) and their accumulation creates a *necrotic core* in the center of the tumor. During this stage, the tumor tends to adopt a spherical shape in which outer-perimeter cells continue to proliferate, while cells in the middle are in a resting state (quiescent). As necrosis and proliferation balance each other, the tumor reaches a limit size (approximately 1-3 mm in diameter).

- *Tumor-induced angiogenesis:* In this stage, cells from the avascular tumor mass induce modifications in the existing *vascular structure* to create new blood vessels that can sustain them. The tumor is able to overcome its limit size, grow much faster and invade the surrounding tissue.

- *Vascular growth and invasive tumor:* This is the most complex stage. The tumor does not form a solid mass anymore and becomes diffusive. Cancer cells can feed on nutrients beyond their immediate vicinity. The formation of necrotic regions becomes much more complex. Tumors do not have a limit size and can grow indefinitely.

In this work, we focus on the *avascular stage* of the tumor and the factors that can provoke its initial growth. We build a simplified model that takes into account key mechanisms identified through a selective review of previous models. We investigate how the model parameters affect the proportion of cancerous cells. Are there any critical values controlling a transition of the tumor from a majority of cancerous cells to a majority of normal ones?

Previous Work

There is a large number of models of tumor growth, covering a great diversity of scales and questions. Byrne (2010) provides a timeline of the most representative models for each stage of tumor growth. Our own brief summary, focusing on the first stage, will not attempt to give an overview but rather highlight rules and parameters deemed important by the literature. An extensive review of models focusing on avascular tumor growth is given by Roose et al. (2007).

Most models fall into two broad categories: *continuum mathematical models*, which are based on partial differential equations and spatial averages; and *discrete cell population models*, which involve individual cell-cell interactions.

Continuum Cell Population Models

These models focus on the relationships between cell density and chemical species that provide nutrients. Typically, they consist of reaction-diffusion and convection equations.

The earliest spatiotemporal and biomechanical models of avascular tumor growth construed the tumor as a 3D "multicellular spheroid" (MCS; Greenspan 1972). Tumor growth was regulated by a single diffusible chemical (oxygen or glucose), supplied externally. The resulting chemical distribution was then taken as a predictor of the underlying spheroid structure, comprising regions of cell proliferation, quiescence and necrosis. However, several simplifications gave these models little applicability: spheroids were assumed to grow radially and symmetrically, there was a single population of cells, and stochastic effects were ignored.

Several modifications and extensions were later brought to the MCS model: relaxing the assumption of radial symmetry, distinguishing different cell populations within the spheroid, and introducing cell movement and pressure (Araujo and McElwain, 2004). One of the most representative works (Casciari et al., 1992) considered a spherical tumor and the effects of certain chemical substances (oxygen, glucose, lactate, and carbon dioxide; bicarbonate, chloride, and hydrogen ions) on cell growth and metabolism. The basic principle at play here is that growth can be limited by chemical diffusion and nutrient consumption. This model also takes into account changes in cell proliferation rate within different chemical environments, and cell movement derived from a law of mass conservation.

Continuum models share several features: they do not distinguish between individual cells, they see tumors as continuous masses, stochastic effects are usually neglected, and subcellular phenomena are ignored. A good review of continuum models and techniques to analyze treatments can be found in (Perthame, 2014).

Discrete Cell Population Models

This category relies on various techniques, such as cellular automata (CA), lattice Boltzmann methods, agent-based modeling, extended Potts, and stochastic approaches (Roose et al., 2007). In all cases, cell state is generally characterized by a multidimensional variable $\mathbf{w} = \{x, v, u\}$, where x is the position of the cell, v its velocity, and u its biological state, which may include its phase in the cell cycle, its interactions with the local chemical environment, and so on.

One of the first discrete models, proposed by Düchting and Vogelsaenger (1985), considered a complex cell cycle model in 3D. Another important study by Qi et al. (1993) used CA rules to reproduce the Gompertz law of cancer growth. Kansal et al. (2000) developed a 3D CA model that did not explicitly include nutrients or mechanical interactions but rather considered proliferation and death rates to be functions of position. "Cellular Potts" approaches, where each biological cell is made up of several lattice points, can also take into account cell membrane tension, cell-cell and cell-matrix adhesion, and chemotaxis (Turner and Sherratt, 2002). Recent years have seen a rise in "hybrid" models, which attempt to combine continuum equations for nutrient concentrations and CA models of cell cycle and migration (Trucu and Chaplain, 2014).

Model

We propose a new model of avascular tumor in its initial stages. The underlying space is a 2D tissue where nutrients are diffusing (Fig. 1). Cells can mutate and become cancerous with a certain probability. They move, proliferate, and die according to the local amount of nutrients, the free space for their offspring, and an internal "energy" that increases with nutrients. Cell positions and cell-cell interactions are constrained to a discrete 50×50 lattice. Cells are represented by circles inside square patches, and are initially assigned a certain position, energy level, and one of three possible types: "normal" (N), "mutated" (M) or "cancerous" (C). A patch can be occupied by one cell only. A few scattered patches coded in red represent blood vessels and can also be occupied by one cell. Despite the presence of blood vessels, this remains a stage-1 avascular tumor simulation since we are not including stage-2 angiogenetic mechanisms. Setup and update rules are explained below.

Agents and Variables

There are two agents in the model: cells and patches. An M cell behaves like an N cell with the difference that it can be affected by its C neighbors. The simulation begins with an initial number of cells, n, and a fixed number of blood-vessel patches, b. Each cell $i = 1, ..., n$ is assigned a state vector $\mathbf{w}_i = (x_i, y_i, e_i, \tau_i, x'_i, y'_i)$ representing its position on the grid, its energy level in $[0, 1]$, its type in {N, M, C}, and free space for its offspring, which can be any unoccupied patch inside a von Neumann domain (4 nearest neighbors) around N and M cells, or a Moore domain (8 nearest neighbors) around C cells. Each patch j on the lattice, whether occupied or not by a cell, is assigned a nutrient level ν_j and a type that can be either "tissue" (T) or "blood vessel" (B).

Parameter	Symbol	Value
Initial number of N cells	n	200
Fixed number of B patches	b	30
Diffusion weight	d	0.8
Unit of nutrient consumption	c	0.02
Division energy of N, M cells	E_N	0.7
Division energy of C cells	E_C	0.6
Division probability of N, M cells	p_N	0.7
Division probability of C cells	p_C	0.8
Mutation probability in N→M→C	m	0.05
Influence of N, M neighb. on C→M	ϵ_N	0.8
Influence of C neighb. on M→C	ϵ_C	0.4

Table 1: Parameters of the model and their standard values.

Setup

First, the parameters of the model (Table 1) are given initial or constant values. Next, each patch is assigned a random nutrient level ν_j uniformly drawn in $[0, 1]$ and a corresponding green-scale color (Fig. 1). Then, n cells are created and placed randomly on the patches. All cells are initialized to type N and assigned a random energy value e_i uniformly drawn in $[0, 1]$. Finally, b blood-vessel patches are randomly created, possibly overlapping with cells, and colored in red.

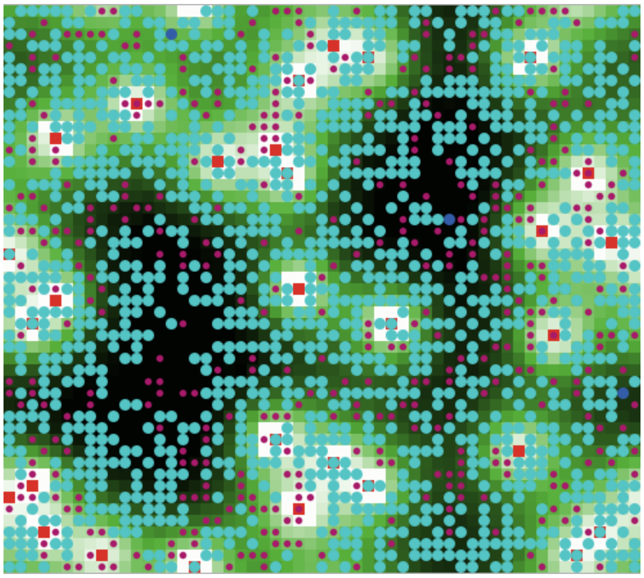


Figure 1: Screenshot of the model after 100 time steps under the parameter values of Table 1. N, M and C cells are colored in blue, cyan and magenta respectively. Patches are in green-scale according to nutrient levels (white for $\nu_j = 1$, black for 0). Here, the tissue contains a large majority of M cells.

Update Rules

At each time step, three rules are applied: cell update, patch update and type update (Fig. 2).

Cell update First, each cell i checks if there are nutrients in the patch j that it occupies, i.e. whether $\nu_j > 0$. If not, the cell dies with probability $1 - e_i$, or if it stays alive, does nothing. If there are nutrients, it consumes a fixed amount c, up to the available quantity ν_j, which increases its energy level accordingly: $e_i \leftarrow e_i + \min(c, \nu_j)$, while the nutrient level of the patch is decreased by the same quantity: $\nu_j \leftarrow \nu_j - \min(c, \nu_j)$. Then, if the cell is cancerous (C), it looks for free space in its 8-patch neighborhood (including blood vessels) and whether it has enough energy to divide, i.e. $e_i > E_C$. This leads to four different scenarios:

- *Free space, enough nutrients:* The C cell divides with probability $e_i * p_C$ and, if it divides, reduces its energy by the cost of division: $e_i \leftarrow e_i - E_C$. Then, it splits this remaining energy into two, keeping one half $e_i/2$ and giving the other half to its daughter.
- *Free space, not enough nutrients:* The C cell moves randomly to one of the unoccupied 8 neighboring patches (including on top of a blood vessel).
- *No free space, enough nutrients:* The C cell waits.
- *No free space, not enough nutrients:* The C cell dies with probability $1 - e_i$, or does nothing.

The same rules apply to noncancerous cell types N and M, except that they use a 4-patch neighborhood, a division energy E_N, and a division probability $e_i * p_N$. Moreover, they wait instead of moving in the second case.

Patch update If the patch j is a blood vessel, then it replenishes its 8 nearest neighbors k with maximum nutrient levels: $\nu_k \leftarrow 1$. After that, nutrients diffuse around all patches with a "diffusion weight" $d \in [0, 1]$. It means that a portion d of each patch's nutrient level is equally distributed over its 8 nearest neighbors: $\nu_k \leftarrow \nu_k + (d/8) * \nu_j$.

Type update (i) If a cell is type N, it can turn into type M with fixed probability m, then it waits until the next time step. (ii) If it is type M, it can turn into type C with the same probability m. Then, if it is still type M, it can turn again into type C, this time with probability $\epsilon_C * n_i^C / n_i$, where n_i is the number of neighboring patches (among 8) occupied by a cell and n_i^C is the number of type-C neighbors. After that, it waits. (iii) If a cell is type C, it can revert to type M with probability $\epsilon_N * n_i^N / n_i$, where n_i^N is the number of type-N and type-M neighbors counted together. Coefficients ϵ_N and ϵ_C are referred to as the "influence of neighbors".

Rationale

Variables and rules are chosen to reflect a realistic, yet simplified, view of tumor development. Ignoring the detailed biochemical activity inside each cell, our intention is to focus on a few key features that affect tumor growth. To represent the competitive advantage that cancerous cells have

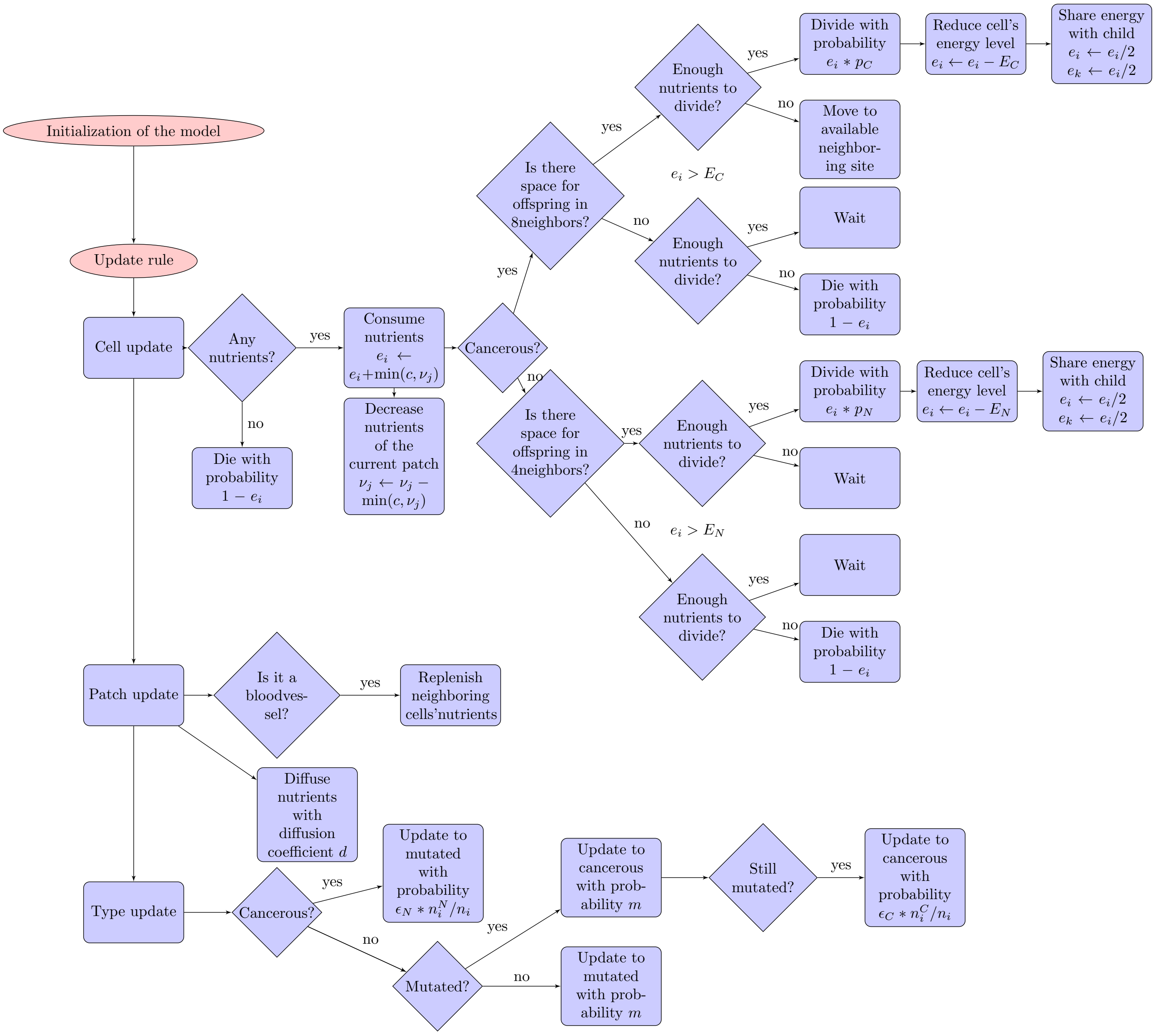

Initialization of the model
Update rule
Cell update
Any nutrients?
yes
no
Die with probability $1 - e_i$
Consume nutrients $e_i \leftarrow e_i + \min(c, \nu_j)$
Decrease nutrients of the current patch $\nu_j \leftarrow \nu_j - \min(c, \nu_j)$
Cancerous?
yes
no
Is there space for offspring in 8neighbors?
yes
no
Enough nutrients to divide?
yes
no
$e_i > E_C$
Divide with probability $e_i * p_C$
Reduce cell's energy level $e_i \leftarrow e_i - E_C$
Share energy with child $e_i \leftarrow e_i/2$ $e_k \leftarrow e_i/2$
Move to available neighboring site
Enough nutrients to divide?
yes
no
Wait
Die with probability $1 - e_i$
Is there space for offspring in 4neighbors?
yes
no
Enough nutrients to divide?
yes
no
Divide with probability $e_i * p_N$
Reduce cell's energy level $e_i \leftarrow e_i - E_N$
Share energy with child $e_i \leftarrow e_i/2$ $e_k \leftarrow e_i/2$
Wait
$e_i > E_N$
Enough nutrients to divide?
yes
no
Wait
Die with probability $1 - e_i$
Patch update
Is it a bloodvessel?
yes
Replenish neighboring cells'nutrients
Diffuse nutrients with diffusion coefficient d
Type update
Cancerous?
yes
no
Update to mutated with probability $\epsilon_N * n_i^N / n_i$
Mutated?
yes
no
Update to cancerous with probability m
Update to mutated with probability m
Still mutated?
yes
Update to cancerous with probability $\epsilon_C * n_i^C / n_i$

Figure 2: Flowchart of the initialization and update rules.

over noncancerous cells, we set their division probability to a larger value ($p_C > p_N$), their corresponding energy cost to a smaller value ($E_C < E_N$), and the space that their offspring can potentially occupy to 8 patches instead of 4. When nutrients are low, C cells can also move to a free patch, as opposed to just waiting like N and M types.

Our introduction of an abstract M type is meant to provide an intermediate type between N and C cells. The idea is that M cells are "carriers of the disease" but still behave like normal cells. They represent the possibility for cancerous cells to revert to a certain form of normalcy (observed experimentally), yet without fully returning to the original N type, so they are still susceptible to their neighbors' influence. One could say that mutated cells are similar to the hypothesis of "dormant" cells in avascular tumors (Udagawa et al., 2002).

We also include the energy as a factor in the actual probabilities of division, $e_i * p_N$ and $e_i * p_C$, to increase the spread of cells that have more energy, or had enough energy for a while and were only waiting for neighboring space to open up. Finally, concerning the influence of neighbors, it is important to note that if $\epsilon_N = 1$ and $\epsilon_C = 0$, there would be only mutated cells at the end of the simulation, whereas in the opposite case, there would be only cancerous cells. Therefore these two parameters must be carefully adjusted to represent a realistic situation. This is one of the topics of the parametric study that follows.

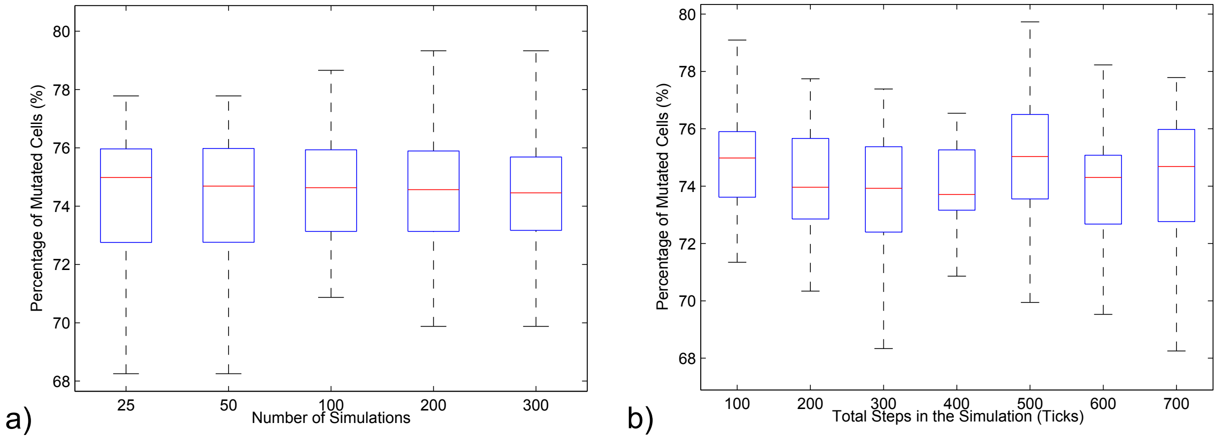


Figure 3: Sensitivity analysis of the model. (a) Average density of mutated cells, $r_M = n_M/n$, calculated over various numbers of simulations lasting 700 time steps each. (b) Same average density calculated over 100 simulations lasting various numbers of time steps. All other parameters are set to the standard values of Table 1. The mean and standard deviation of r_M change only little, at around $\overline{r_M} \approx 74.5\%$ and $\Delta r_M \approx 2.5\%$, indicating that the behavior of the model is consistent across experiments.

Results

Measurements

To analyze the dynamics of the model, we measure the overall densities and "clustered" densities of normal, mutated and cancerous cells. An N cell is said to belong to a cluster if there are at least 5 other N cells in its 8-patch neighborhood (same for M and C cells).

Sensitivity Analysis

Since the model is stochastic, we must evaluate the number of simulations S and the number of time steps per simulation T needed to conduct a parameter space exploration that is sufficiently representative of the generic model behavior. To carry out this sensitivity analysis, we use the parameters of Table 1 and calculate the overall density of M cells, $r_M = n_M/n$. Setting $T = 700$ under various numbers of simulations $S = 25, ..., 300$ (Fig. 3a), we observe that the range of r_M (mean and variance) is about constant for $S \geq 100$. Conversely, setting $S = 100$ under various durations $T = 100, ..., 700$ (Fig. 3b) shows that r_M tends to stabilize for $T \geq 500$. Therefore, we conclude that the behavior our model is robust and adopt $S = 100$ and $T = 500$ to keep the computing cost reasonable in the rest of the study, since the distributions of r_M are similar above these values.

Parameter Space Exploration

To explore the dynamical regimes of the model, we select the following dimensions of parameter space, expected to have an important impact on the outcome: the influence of neighbors, ϵ_N and ϵ_C; the division probability of noncancerous cells, p_N; and the mutation probability, m. In each case, a pair of values or a single value is varied, while all other parameters are kept at the standard settings of Table 1.

Influence of neighbors We systematically vary ϵ_N and ϵ_C in the interval $[0, 1]$ by increments of 0.1, for a total of 11×11 experimental points, and plot the mean and standard deviation of the density of M cells and C cells (Fig. 4). The mean density of N cells consistently remains under 3% and is not shown. This reveals a phase transition from a majority of C cells to a majority of M cells along a boundary line roughly at $\epsilon_N = \epsilon_C + 0.2$. Therefore, to push a cancerous tissue back into a majority of M cells, the mutating influence of N and M cells has to be significantly greater than that of C cells.

Division probability We choose two points in the high M-cell density domain of the neighbors' influence space (Fig. 4a): $A = (\epsilon_N, \epsilon_C) = (0.8, 0.1)$ and $B = (0.8, 0.4)$ (the standard values), and vary the division probability of N cells p_N in $[0, 0.7]$ by increments of 0.1. We observe that in point A, p_N does not have much effect on the dynamics in the model, which invariably ends with many more M cells than C cells, about 85% vs. 15%, while C cells remain isolated (Fig. 5a,c). In point B, by contrast, pulling p_N below 0.2 makes the densities of M and C cells converge to comparable values, about 50% overall and 15-20% in clusters (Fig. 5b,d). It means that to stay outside the high cancerous density domain, N and M cells should have a greater mutating influence than C cells on their neighbors (previous conclusion), or N and M cells should divide reasonably fast.

Random mutation probability With the standard parameters of Table 1, we vary the cancerous neighbors' influence ϵ_C under three different basic N→M→C mutation probabilities. Under the standard low value $m = 0.05$, we find again a transition between high M-cell and high C-cell density domains at $\epsilon_C \approx 0.6$ (Fig. 6a). Under a higher value $m = 0.15$, the transition happens earlier at $\epsilon_C \approx 0.2$ (Fig. 6b). Under very high mutation rate $m = 0.25$, the neighbors' influence

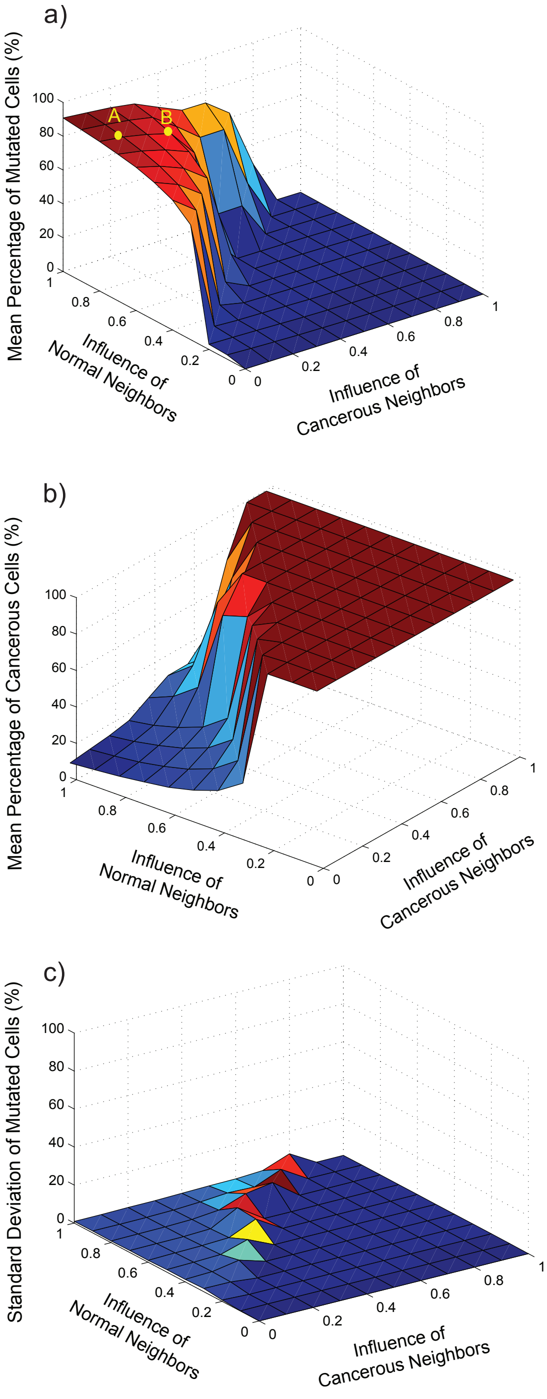


Figure 4: Exploring the influence of neighbors. Both mutating factors, ϵ_N and ϵ_C, are varied in $[0, 1]$ while the other parameters are as in Table 1. (a) Mean density of mutated cells, $\overline{r_M}$. (b) Mean density of cancerous cells, $\overline{r_C}$. (c) Standard deviation of density of M cells, Δr_M (same as Δr_C, not shown here). All graphs exhibit a clear phase transition along a boundary line roughly at $\epsilon_N = \epsilon_C + 0.2$. Parametric point $B = (\epsilon_N, \epsilon_C) = (0.8, 0.4)$ in the high M-cell density domain is the standard. The other point, $A = (0.8, 0.1)$, will be used when varying the division probability p_N (Fig. 5).

is ignored by the system: almost all cells are cancerous and no transition back to a mutated state is possible. (Fig. 6c).

Importance of Rules

We also analyze the importance of certain rules by measuring the effects of "rule knockout" on our model: (i) removing energy-dependency from division probabilities and setting them to constant rates p_N and p_C; (ii) conversely, removing type-specificity from division probabilities and setting them to the energy level e_i for all cells; (iii) finally, removing type-specificity from division space and giving all cells the same 8-patch Moore neighborhood to spawn offspring. In each scenario, the neighbors' influence parameter space $(\epsilon_N, \epsilon_C) \in [0, 1]^2$ is explored again and the transition area between high M-cell and high C-cell density domains is estimated and plotted in 2D (Fig. 7). A parametric point is considered to lie inside the transition area if the variance of the M-cell density, $(\Delta r_M)^2$, is greater than the mean variance over all 121 points. We observe that scenarios (i) and (iii) provoke a significant displacement of the transition area (Fig. 7b,d), whereas scenario (ii) leaves it almost unchanged (Fig. 7c). Therefore, we conclude that energy-dependency of division and offspring space are key mechanisms, while type-specificity of division is not.

Discussion

We presented a simple agent-based model of tumor growth able to give rise to two different situations: one where the majority of cells is cancerous and another one where it is mutated. We showed the outcomes of varying the influence of neighbors, division probabilities and mutation probability. A constant feature of the system's behavior is that most normal cells disappear and only a competition between mutated and cancerous cells remains (Fig. 1). The biological conclusion is that, in the end, all normal cells become influenced by the type of their neighbors. We could expect normal cells to survive longer, however the rules and standard parameter values of our model give a clear advantage to cancerous cells across all dimensions (more space for the offspring, less division energy, higher probability of dividing) and lead to a rapid depletion of normal cells.

Our exploration of neighbors' influence parameter space revealed a phase transition area, inside which the standard deviation of cell densities is clearly positive, i.e. the outcomes of simulations vary significantly more (Fig. 4). This implies the existence of two types of parametric points in the model: ones such as A which are far away from the phase transition and create a stable system, and ones such as B which are close to the phase transition and create instability. This was illustrated by varying the division probability in point B and showing that the density of mutated cells could drop below that of cancerous cells (Fig. 5b,d). Another transition between highly mutated and highly cancerous states was found along the mutation probability axis at

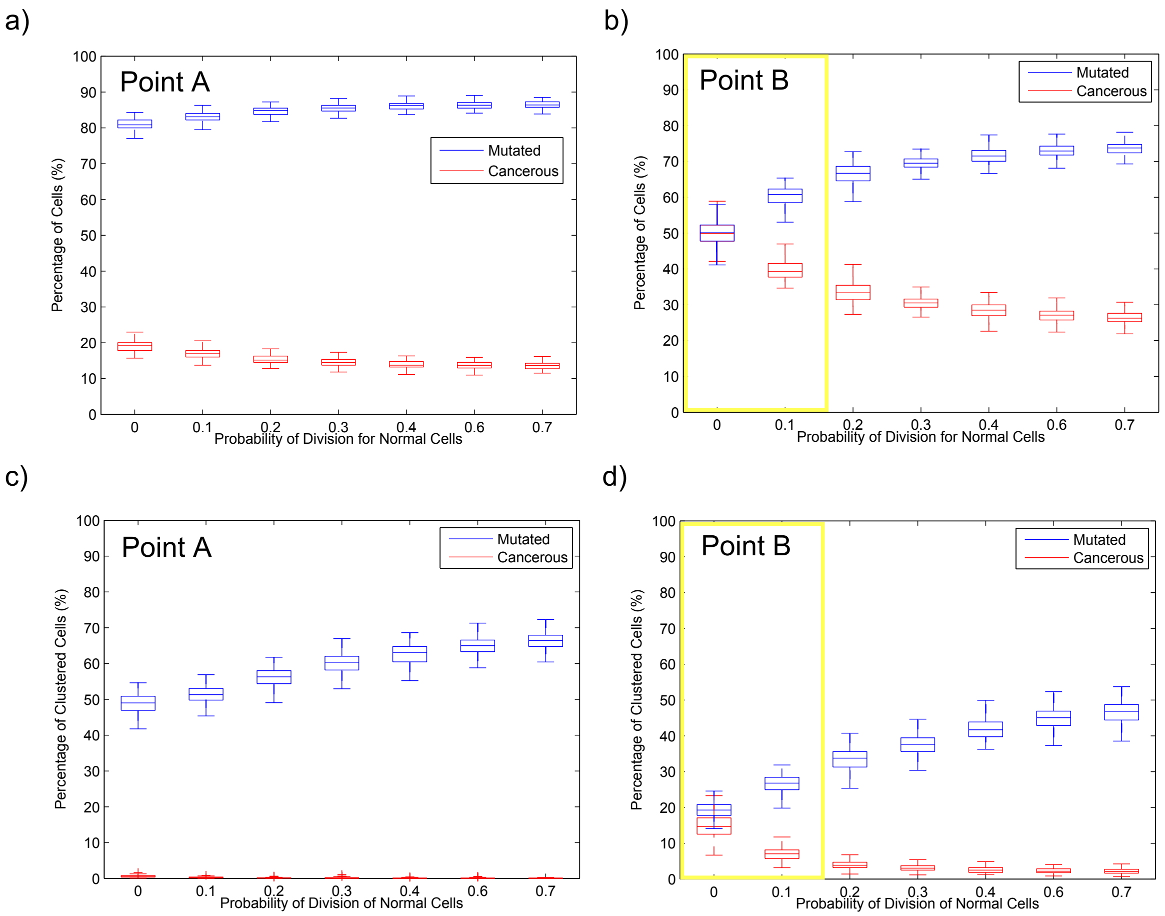


Figure 5: Exploring the division of noncancerous cells. The probability p_N is varied while other parameters are as in Table 1, except the pair (ϵ_N, ϵ_C) which is set to point A or point B as in Fig. 4a. (a,b) Average overall densities of mutated cells (in blue) and cancerous cells (in red). (c,d) Average densities of clustered M and C cells. (a,c) Parametric conditions A: p_N has little effect on the outcome, characterized by a significant majority of M cells and isolated C cells. (b,d) Standard parametric conditions B: low values of p_N (highlighted in yellow) lead to comparable numbers of M and C cells.

around 20%, above which only C cells remained (Fig. 6).

By varying these parameters, we examined both internal and external factors of tumor growth. In future work, the diffusion weight d will be another important parameter to explore, in order to check whether growth can be limited by diffusion as it was the case in earlier models of avascular tumors. Finally, by analyzing the effect of rule knockout on the transition curve we were able to assess the significance or neutrality of certain rules in our model.

Compared to our work, other agent-based models in the literature generally contain more complex and realistic descriptions of cellular agents and tumor environment (review in Wang et al., 2015). They strive to take into account multiple phenotypic features such as molecular signaling, cellular metabolism and other mutation-induced changes (e.g. Ramis-Conde et al., 2008). By contrast, our goal was to identify a minimal set of rules enabling behaviors typical of avascular tumor growth: proliferation in nutrient-rich regions, limitation in size, and phase transitions. This also allowed us to conduct a parameter space exploration along a few dimensions and assess the effects of varying key values.

Naturally, our model can be improved in several ways. For example, in order to minimize the processing time, we assumed that nutrients diffused at constant speed regardless of patch occupancy. In reality, proteins, lipids and other particles undergo *anomalous* diffusion due to molecular crowding (Banks and Fradin, 2005). Second, instead of setting the parameters to empirical values appropriate for the numerical simulations, one could try to match them to real imaging datasets and measurements from specific tumors, for example via a “fitness” function evaluating the realism (overlap) between the digital and the physical object.

In sum, we tried to describe the dynamics of the initial stages of avascular tumor growth using only key principles and concepts without including chemical or mechanical details. Future extensions of this model should take into ac-

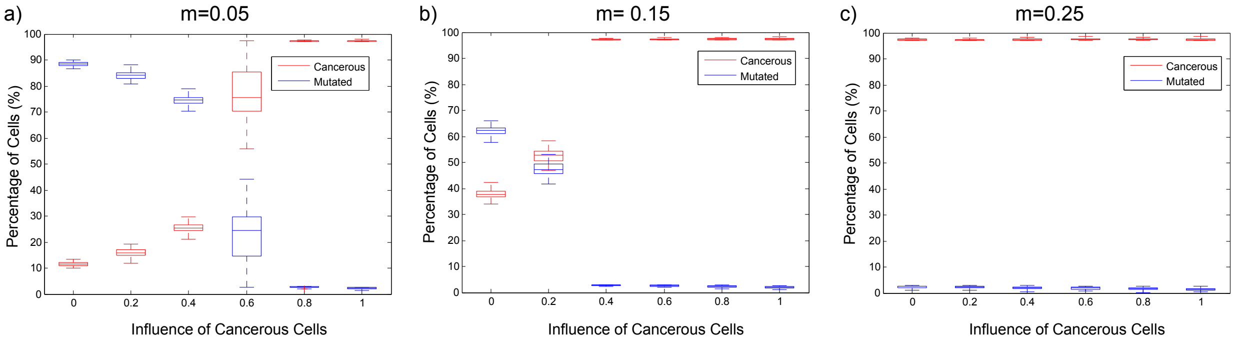


Figure 6: Exploring the mutation probability m. The cancerous neighbors' influence ϵ_C is varied while other parameters remain as in Table 1. (a) At the standard, low m value, we obtain the same transition as Fig. 4. (b,c) At higher m values, the transition shifts in favor of all-cancerous end states.

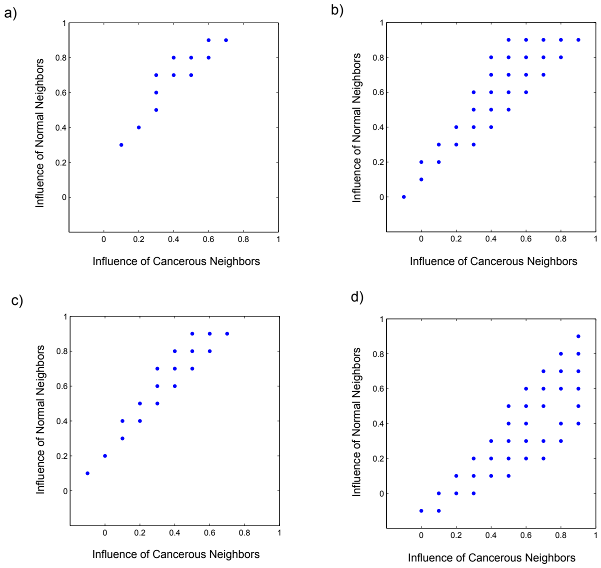


Figure 7: Importance of rules. The transition curve in (ϵ_N, ϵ_C) space can be more or less shifted in different versions of the model. (a) Standard rules: curve as in Fig. 4. (b) No dependence on energy in the division probabilities → significant shift. (c) No distinction between types in the division probabilities → no significant shift. (d) More neighborhood space for noncancerous cells → significant shift.

count both aspects as well as the cell cycle. Moreover, they should address the possibility of fitting the parameters to clinical data about the delimitation of the proliferating region, the location of the phase transition, and the limit size of the avascular tumor.

References

Araujo, R. and McElwain, D. (2004). A history of the study of solid tumour growth. *Bull. Math. Biol.*, 66(5):1039–1091.

Banks, D. S. and Fradin, C. (2005). Anomalous diffusion of proteins due to molecular crowding. *Biophysical Journal*, 89(5):2960–2971.

Byrne, H. M. (2010). Dissecting cancer through mathematics: from the cell to the animal model. *Nature Reviews Cancer*, 10(3):221–230.

Casciari, J., Sotirchos, S., and Sutherland, R. (1992). Mathematical modelling of microenvironment and growth in emt6/ro multicellular tumour spheroids. *Cell Proliferation*, 25(1):1–22.

Düchting, W. and Vogelsaenger, T. (1985). Recent progress in modelling and simulation of three-dimensional tumor growth and treatment. *Biosys.*, 18(1):79–91.

Greenspan, H. (1972). Models for the growth of a solid tumor by diffusion. *Stud. Appl. Math*, 51(4):317–340.

Kansal, A., Torquato, S., Chiocca, E., and Deisboeck, T. (2000). Emergence of a subpopulation in a computational model of tumor growth. *J. Theo. Biol.*, 207(3):431–441.

Perthame, B. (2014). Some mathematical aspects of tumor growth and therapy. In *ICM 2014*.

Qi, A.-S., Zheng, X., Du, C.-Y., and An, B.-S. (1993). A cellular automaton model of cancerous growth. *J. Theo. Biol.*, 161(1):1–12.

Ramis-Conde, I., Drasdo, D., Anderson, A. R., and Chaplain, M. A. (2008). Modeling the influence of the e-cadherin-β-catenin pathway in cancer cell invasion: a multiscale approach. *Biophysical journal*, 95(1):155–165.

Roose, T., Chapman, S. J., and Maini, P. K. (2007). Mathematical models of avascular tumor growth. *Siam Review*, 49(2):179–208.

Stewart, B. W. and Wild, C. P. (2014). *World cancer report 2014*. IARC Press, International Agency for Research on Cancer.

Trucu, D. and Chaplain, M. A. (2014). Multiscale analysis and modelling for cancer growth and development. In *Managing Complexity, Reducing Perplexity*, pages 45–53. Springer.

Turner, S. and Sherratt, J. A. (2002). Intercellular adhesion and cancer invasion. *J. Theo. Biol.*, 216(1):85–100.

Udagawa, T., Fernandez, A., Achilles, E.-G., Fokman, J., and DAmato, R. J. (2002). Persistence of microscopic human cancers in mice: alterations in the angiogenic balance accompanies loss of tumor dormancy. *The FASEB journal*, 16(11):1361–1370.

Wang, Z., Butner, J. D., Kerketta, R., Cristini, V., and Deisboeck, T. S. (2015). Simulating cancer growth with multiscale agent-based modeling. *Seminars in Cancer Biology*, 30:70–78.

Spatial Computing in Synthetic Bioware: Creating Bacterial Architectures

Jonathan Pascalie[1,2], Martin Potier[2], Taras Kowaliw[1], Jean-Louis Giavitto[3],
Olivier Michel[2], Antoine Spicher[2] and René Doursat[1,4]

[1]Complex Systems Institute, Paris Île-de-France (ISC-PIF), CNRS (UPS3611); [2]LACL, Université Paris-Est Créteil, France
[3]IRCAM, CNRS (UMR9912), Paris, France; [4]BioEmergences Lab, CNRS (USR3695), Gif-sur-Yvette, France

Synthetic biology is an emerging scientific field that promotes the standardized manufacturing of biological components without natural equivalents. Its goal is to create artificial living systems that can meet various needs in health care, nanotechnology and energy. Most works are currently focused on the individual bacterium as a chemical reactor. Our project, SynBioTIC, addresses a novel and more complex challenge: *shape engineering*, i.e. the redesign of natural morphogenesis toward a new kind of "developmental 3D printing". Potential applications include organ growth, natural computing in biocircuits, or future vegetal houses. Using realistic agent-based simulations of bacterial mats, we experiment with mechanisms allowing cell assemblies to collectively self-repair and develop complex structures.

To create multicellular organisms that exhibit specific shapes (a completely original task) we construe their development iteratively by combining basic processes such as homeostasis, segmentation, and controlled proliferation *in silico*. We use the *E. coli* simulator Gro[1], a physico-chemical computation platform offering reaction-diffusion and collision dynamics solvers. The synthetic "bioware" of our model executes a set of rules, or "genome", in each cell. Cells can differentiate into several predefined *types* associated with specific *actions* (divide, tumble, emit signal, die, etc.). Transitions between types are triggered by *conditions* involving internal and external sensors that detect various protein levels inside and around the cell. There is no direct molecular signaling between two neighboring bacteria, only indirect communication via morphogen diffusion and the mechanical constraints of 2D packing. In any case, the overall architecture emerges in a purely endogenous fashion.

For now, cell behaviors are set by rules hand-coded in the Gro script language. Starting from a single bacterium, our artificial creatures execute a series of developmental stages. First, isotropic proliferation produces a roughly circular population characterized by homeostatic activity (black and white cores in Fig. 1a,b). This is based on leader cells emitting a morphogen, while other cells continually divide and die at the periphery where the morphogen concentration drops. Then, the central region of the disc differentiates from the crown. Each cell also contains an oscillatory mechanism acting like a internal clock. In the crown, these oscillators are synchronized, i.e. characterized by a uniform phase. At this stage, a new wave of morphogen is triggered by a randomly activated cell on the crown, and rapidly propagates (suppressing any competitor wave). The encounter between the wave front and the current state of the oscillator determines whether each cell differentiates, and into what type. The period of oscillations controls the number of segments that can appear. Finally, precursor cells emerge at the periphery of these segments and stimulate new local proliferation, which eventually triggers limb growth in a way similar to the apical meristem of plant shoots (Fig. 1c).

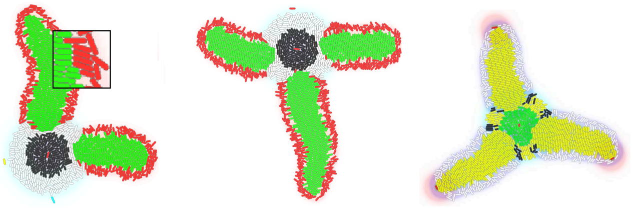


Figure 1: Example of simulated organisms. (a,b) T and L shapes. Each limb of the organism stems from differentiated precursor cells. Starting from a four-pointed star, this makes it possible to introduce a "divergence of homology" through different parameter values in each limb, whether unequal lengths or complete silencing. (c) Three-pointed star shape. Here, limb growth is undifferentiated and the organism exhibits radial symmetry.

Applying this mechanism to two segments and two precursors, North and South, then on the equator that they form (areas of equal morphogen concentration), the system gives rise to a second pair, East and West, i.e. four differentiated seeds in total. This makes it possible to control the growth and features of single appendages. The L and T shapes of Fig. 1 exemplify this "divergence of homology": some precursors are inhibited while others create limbs of varying size. Such morphogenetic phenotypes allow us to envision more complex shapes made of an array of cores and limbs, by iterating the above processes. Most importantly, they open the door to an evolutionary ("evo-devo") exploration.

[1] Jang, Oishi, Egbert & Klavins (2012) *ACS Syn Biol*, 1:365–74.

Do it yourself (DIY) liquid handling robot for evolutionary search exploration

Asbjørn Müller[1], Aleksandra Amaldass[1] Kliment Yanev[2] and Steen Rasmussen[1,3]

[1]Center for Fundamental Living Technology (FLinT), University of Southern Denmark, 5230 Odense, Denmark
[2]Future Bits Open-Tech UG, Germany, [3]Santa Fe Institute, Santa Fe 87501, New Mexico, USA
steen@sdu.dk

Open source hardware has made it significantly easier to build custom scientific instrumentation, in particular in the recent wake of open-source 3D printing technology (Jones et al., 2011). The main advantages of DIY scientific instruments include (i) significantly lower instrumentation costs, (ii) possibility for open-ended special purpose functions to address research-specific problems, (iii) deeper understanding of the scientific problem under investigation as the instrumentation is designed simultaneously, as well as (iv) providing an excellent student learning tool. We present the ChemBot: a programmable liquid handling robot, which is able to realize a series of desired functions and processes.

The goal is to build an inexpensive, open source, open hardware, liquid handling robot capable of automated chemical, biophysical or molecular biology/biochemistry experiments - including photochemical reactions. Ideally, it should be programmable for multiple parameters and parallel experimentation, including implementation of evolutionary search algorithms; and also incorporate optical observational capabilities to document, survey and analyze experiments.

Features of the ChemBot include deposition of liquids that are pre-loaded into a syringe by semi-automated action; it is capable of simultaneous delivery of up to three different liquids into single micro wells on a 96-well plate, and agitate the contents of the individual wells. Programmable/automatic surveillance by photographic documentation is possible via a microscope. A LED array is present for individual illumination of the wells from below with UV, blue, and white light. The design is highly hackable (easily modyfiable and adaptable) and uses syringe pumps ranging from 1 to 20 ml syringes. The control software can be adapted to most experimental design processes.

Preliminary results. Delivery of volumes as small as 80 nl has been tested and reliability has been observed with volumes as small as 0.5µl (0.5‰ of a total syringe capacity of 1 ml water). Precision and reliability of liquid delivery is the first challenge to overcome (Gilson, 2007).

Combinatorial design of experiments. The ChemBot is designed to support Predictive Design Technology (Caschera et al., 2010), an iterative procedure to efficiently discover desired experimental properties by means of visual (microscope) inspection in extremely large experimental parameter spaces.

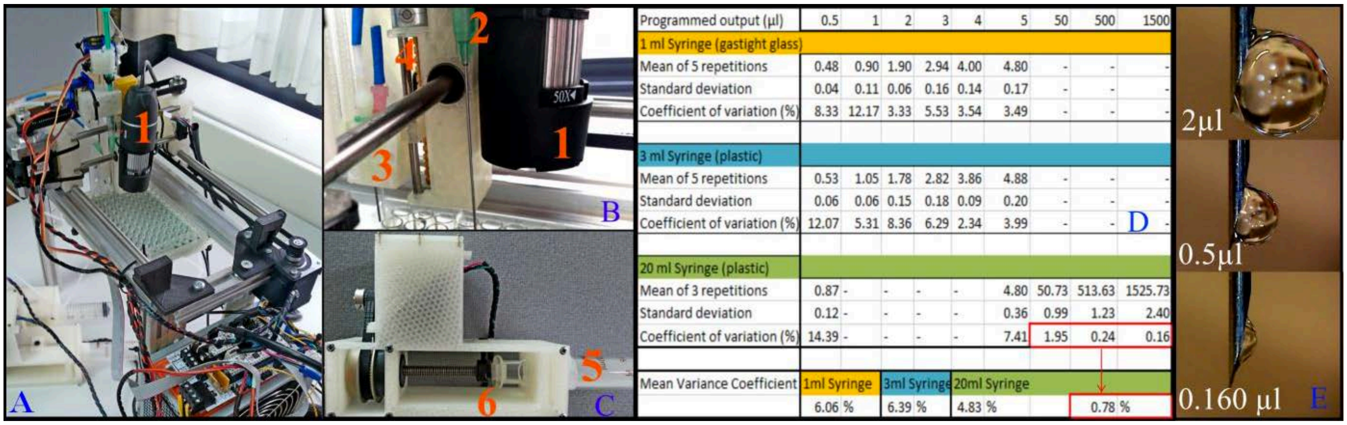
Acknowledgements. This work is in part funded by the European Commission TRUCE and MICREAgents projects.

References

Cashera et al. (2010) DOI: 10.1371journal.pone.0008546

Gilson, Verification Procedure for Accuracy and Precision, 2007, Procedure LT802292/F

Jones et al. (2011) *Robotica* **29**, 177-191



Programmed output (µl)	0.5	1	2	3	4	5	50	500	1500
1 ml Syringe (gastight glass)									
Mean of 5 repetitions	0.48	0.90	1.90	2.94	4.00	4.80	-	-	-
Standard deviation	0.04	0.11	0.06	0.16	0.14	0.17	-	-	-
Coefficient of variation (%)	8.33	12.17	3.33	5.53	3.54	3.49	-	-	-
3 ml Syringe (plastic)									
Mean of 5 repetitions	0.53	1.05	1.78	2.82	3.86	4.88	-	-	-
Standard deviation	0.06	0.06	0.15	0.18	0.09	0.20	-	-	-
Coefficient of variation (%)	12.07	5.31	8.36	6.29	2.34	3.99	-	-	-
20 ml Syringe (plastic)									
Mean of 3 repetitions	0.87	-	-	-	-	4.80	50.73	513.63	1525.73
Standard deviation	0.12	-	-	-	-	0.36	0.99	1.23	2.40
Coefficient of variation (%)	14.39	-	-	-	-	7.41	1.95	0.24	0.16
Mean Variance Coefficient	1ml Syringe		3ml Syringe		20ml Syringe				
	6.06 %		6.39 %		4.83 %			0.78 %	

Figure 1 (**A**) A system of smooth rods with mounted carriages enables two-dimensional movement of the tool assembly. (**B**) The latter consists of (1) a microscope, (2) a mixing syringe, and (3) a syringe tip retainer, actuated in the z direction by (4) a spindle, which in turn is revolved by a small stepper motor. (**C**) Liquid is delivered through the tips connected via tubes to the syringe pumps. The syringe pumps push directly on (6) the piston (of (5) the syringe) by linear motion of a threaded rod, gear reduced by a belt and pulley assembly. (**D**) Table of deposition accuracy over a range of volumes (0.5 - 1500 µl) using three different syringes. (**E**) Different droplet volumes in the lower range using the 1ml glass syringe.

Human Swarms, a real-time method for collective intelligence

Louis B. Rosenberg

Unanimous A.I., San Francisco, California, USA
Louis@UnanimousAI.com

Overview

Although substantial research has explored the emergence of collective intelligence in real-time human-based collaborative systems, much of this work has focused on rigid scenarios such as the Prisoner's Dilemma (**PD**). (Pinheiro et al., 2012; Santos et al., 2012). While such work is of great research value, there's a growing need for a flexible real-world platform that fosters collective intelligence in authentic decision-making situations. This paper introduces a new platform called **UNUM** that allows groups of online users to collectively answer questions, make decisions, and resolve dilemmas by working together in unified dynamic systems. Modeled after biological swarms, the **UNUM** platform enables online groups to work in real-time synchrony, collaboratively exploring a decision-space and converging on preferred solutions in a matter of seconds. We call the process *"social swarming"* and early real-world testing suggests it has great potential for harnessing collective intelligence.

Background

Humanity is a tribal species, owing our evolutionary success to our ability to collaborate in social groups (Axelrod, 1981; Rand et al., 2011). This said, modern life has expanded the scope of human interactions so widely, our tribal norms may no longer be sufficient to maintain a cooperative stance among dependent parties (Green, 2013). Even among small social groups, collaborators rarely congregate in the same place at the same time, decisions often being made via email and text. For larger groups, *discussion forums* are commonly used for distributed online decisions, with conclusions based on asynchronous user inputs such as "likes" and "up-votes". Unfortunately, asynchronous polling doesn't leverage our natural capacity for compromise and consensus-building. In fact, recent studies suggest that asynchronous polling, as used by mainstream social media sites and forums, greatly distorts group-wise decisions by introducing biasing effects known commonly as *herding* or *snowballing* (Muchnik et al., 2013).

From Polls to Swarms

As introduced above, there is a growing need for new online platforms that facilitate collective intelligence and support collaborative decision-making without employing traditional asynchronous polling. To address this, we developed UNUM, a real-time collective intelligence engine that is modeled after natural biological swarms. UNUM enables groups of users to answer questions *in synchrony*, the participants working as a unified dynamic system through real-time feedback loops.

When using the UNUM platform, swarms of online users can answer questions and make decisions by collaboratively moving a graphical **puck** to select among a set of possible answers. The puck is generated by a central server and modeled as a real-world physical system with a defined mass, damping and friction. Each participant in the swarm connects to the server and is provided a controllable graphical **magnet** that allows the user to freely apply force vectors on the puck in real time (Fig. 1). The puck moves in response to swarm's influence, not based on the input of any individual participant, but based on a dynamic feedback loop that is closed around all swarm members. In this way, real-time *synchronous control* is enabled across a swarm of distributed networked users.

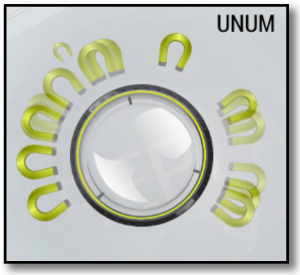


Figure 1: a human swarm of user-controlled magnets collaborate in synchrony to move a graphical puck as a unified collective intelligence.

Through the collaborative control of the graphical puck, a real-time *physical negotiation* emerges among the members of the online swarm. This occurs because all of the participating users are able to push and pull on the puck at the same time, collectively exploring the decision-space and converging upon the most agreeable answers. But do the answers have value?

Early Testing and Results

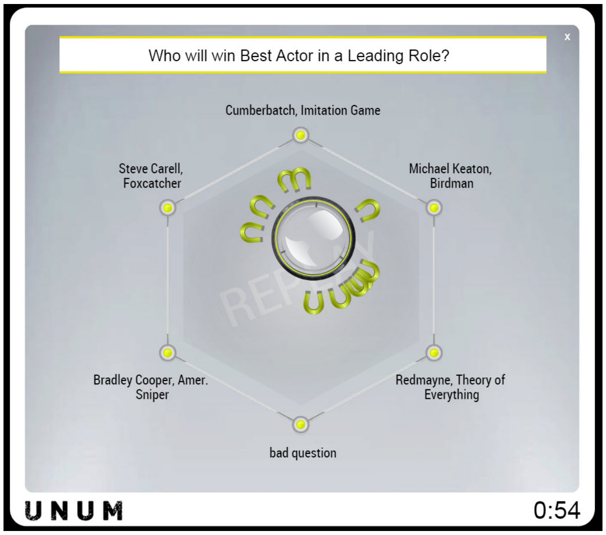
To test the value of human swarms, we enlisted groups of novice users and asked them to make predictions on verifiable events: the outcome of the NFL playoffs, the Golden Globes, and the 2015 Academy Awards. In all cases, the predictions made by swarms were substantially more accurate than the predictions made by the individuals who comprised each swarm. In fact, in all cases the predictions made by swarms out-performed even the highest performing individual in each group. The swarms also out-performed the average polling results across the full population of participants. This suggests that swarms offer a powerful alternative to the traditional poll-based methods of harnessing the wisdom of groups.

For example, when predicting the 2015 Academy Awards, we polled 48 individuals with a written survey, asking them to predict the top 15 award categories. Using the most popular predictions to represent "the wisdom of the crowd", the group collectively achieved **6 correct predictions** for the top 15 award categories (40% success). This was our baseline dataset, the low success rate reflecting the fact that this group of users had no special knowledge about movies.

To test swarming, we then selected a 7 person sub-group of the full population and asked them make the same predictions, but now as a unified dynamic system. The 7 individuals were typical performers on the written poll, ensuring equity with the full 48 person population. Each of the 7 individuals were networked over standard internet connections to a central server from different remote locations.

Working as a unified swarm, the group of 7 individuals achieved **11 correct predictions** for the top 15 award categories (73% success). In other words, a sub-group that was only 15% the size of the full population had a success rate that was nearly double. We believe this is a highly promising result and speaks to the potential for harnessing the wisdom of social groups through real-time swarming.

It should also be noted that real-time swarming is a high-speed process, all decisions made within 60 seconds or less. Thus, in addition to improved accuracy of predictions, this form of collective intelligence is uniquely efficient.



Who will win Best Actor in a Leading Role?

Figure 2: screen-shot of a real-time social swarm in the process of predicting the Best Actor category of the 2015 Academy Awards.

As a point of reference, experts at the New York Times made similar predictions for the 2015 Academy Awards. These experts possessed far deeper knowledge than the novice members of our study. We assume these experts invested far more than 60 seconds on each prediction made. Still, the New York Times only showed a 55% success rate.[1] Thus, a group of 7 novices, functioning as a social swarm, made predictions that surpassed industry experts. Although not conclusive, this result suggests that social swarming may provide a means of achieving expert-level insights from groups of non-experts.

Discussion and Conclusions

Why are swarms better than polls? Our analysis suggest that while polls are good at characterizing the average views of a population, without real-time feedback control, polls offer no means for groups to explore options and find consensus. Swarms, on the other hand, allow users to continually update their intent in real-time, assessing how their views combine with the other participants to achieve an acceptable outcome.

In this way, each participant in a swarm is not expressing a singular view, but is continually assessing his own personal conviction across the range of possible options, weighing his confidence and preference in real-time. With all participants doing this in synchrony, the swarm quickly converges on solutions that seem to maximize the collective confidence and preference of the full group. We believe this is why swarms are able so efficiently capture the group's wisdom.

We are currently conducting additional studies to quantify the effectiveness of social swarms, not just to make accurate predictions but in facilitating group decisions. Of particular interest is whether decisions made by real-time swarms are more or less satisfactory to the participants than decisions made by traditional polling. Initial results suggest that social swarms yield more satisfactory decisions than votes or polls.

Finally, to help drive exploration of social swarming, we have made the UNUM platform accessible to any academics who wish to run their own user tests. Academic researcher can request a free account at www.unum.ai

Acknowledgements: This study was funded by Unanimous A.I. with support from California State University (CalPoly, San Luis Obispo).

References

Axelrod R, Hamilton WD (1981) The evolution of cooperation. Science 211:1390–1396.

Greene, Joshua (2013). *Moral Tribes: Emotion, Reason, and the Gap Between Us and Them*. Penguin Press.

Lev Muchnik, Sinan Aral, Sean J. Taylor. Social Influence Bias: A Randomized Experiment. Science, 9 August 2013: Vol. 341 no. 6146 pp. 647-651

Rand, D. G., Arbesman, S. & Christakis, N. A. (2011) Dynamic social networks promote cooperation in experiments with humans. Proc. Natl Acad. Sci. USA 108, 19193–19198.

Pinheiro, F. L., Santos, F. C., and Pacheco, J. M. (2012). How selection pressure changes the nature of social dilemmas in structured populations. New J. Phys., 14(7):073035.

Santos, F. C., Pinheiro, F. L., Lenaerts, T., and Pacheco, J. M. (2012). The role of diversity in the evolution of cooperation. J. Theor. Biol., 299:88–96.

[1] http://carpetbagger.blogs.nytimes.com/2015/02/19/oscars-2015-the-carpetbaggers-predictions/

Author Index